DOCUMENT RESUME

ED 031 267 LI 001 603

By-Divett, Robert T.; Jones, W. Wayne
Report on the Total System Computer Program for Medical Libraries.
New Mexico Univ., Albuquerque. Library of the Medical Sciences.
Spons Agency-National Library of Medicine, Bethesda, Md.
Pub Date May 69
Note-429p.
EDRS Price MF-$1.75 HC-$21.55
Descriptors-*Computer Programs, *Information Systems, Library Networks, *Library Technical Processes, *Man Machine Systems, *Medical Libraries, Systems Development

The objective of this project was to develop an integrated computer program for the total operations of a medical library including acquisitions, cataloging, circulation, reference, a computer catalog, serials controls, and current awareness services. The report describes two systems approaches: the batch system and the terminal system. The batch system is a combination of manual and machine processing of the clerical functions of a library. This particular system involves functions in the acquisitions, cataloging and circulation departments. The terminal system is an on-line computer system with terminals, in each section of the library, including reference services and patron use. It is designed to cover the total operations of a library. After a general description of both systems the report lists the equipment needed and the estimated costs for both. Recommendations are given for sharing the system with several other medical libraries. Technical descriptions of the terminal system, including file organization and program structure, are presented in the report. The appendices show library forms (order forms, catalog cards, etc.), 2260 screen displays, and source listings. (Author/RM)

ED031267

LI 001603

Report on
THE TOTAL SYSTEM COMPUTER PROGRAM FOR MEDICAL LIBRARIES

ERIC/CLIS
AUG 6 '69
UNIVERSITY OF MINNESOTA

THE UNIVERSITY OF NEW MEXICO
SCHOOL OF MEDICINE
LIBRARY OF THE MEDICAL SCIENCES

MAY 1969

LI 001603

Report on

The Total System Computer Program for Medical Libraries

by

Robert T. Divett, Ed. D.
Librarian and Associate Professor of Medical Bibliography
Principal Investigator
and
W. Wayne Jones
Chief, Research and Development Section

Sponsored by the
National Library of Medicine
Grant Number RO 1 LM00053

University of New Mexico
School of Medicine
Library of the Medical Sciences

May 1969

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
OFFICE OF EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL OFFICE OF EDUCATION POSITION OR POLICY.

PREFACE

The increased output of published information in recent years in the medical field has made the concept of automation in the medical libraries a great concern for both the librarian and the data processor. It has made both professions aware of the need for an automated system which will efficiently handle the increased processing of information and, at the same time, simplify the method of retrieval for the user. Up to this time, automated systems have mechanized one aspect or another, but not all of them in one system. The experimental computer program for medical libraries at the University of New Mexico Library of the Medical Sciences is an attempt to design a total computer system.

The results of that research are detailed in this manual. It is hoped it is readable to the librarian to make him fully aware of the capabilities of automation in libraries. It is also hoped it is detailed enough to give the data processor the information he will need to install the system. This is accomplished by combining the two approaches in one, more general statement in Part I and by giving the technical details and descriptions in Part II. This divided format makes the information understandable and available to both the librarian and the data processor.

ACKNOWLEDGMENTS

The research represented by this study reflects the efforts of many people who were assigned to the project at one time or another. The authors would like to recognize these members for their many contributions.

Mr. Edwin S. Dahl, Systems Programmer, must be commended for his valuable assistance. His work in writing the specialized assembler language programs for program inter-communication was especially helpful.

Special acclaim must be given to Mr. S. S. Millwright, Systems Programmer, for his efforts in constructing the randomly accessable MeSH tree structure. He also performed yeoman duties in other project areas.

The remaining members of the project staff, Mr. James Huang and Mr. Lin Lewis, Programmers, we thank for producing many of the processing modules.

Particular consideration should be given Mr. Robert Botts, I.B.M.; Mr. Larry Unkrich and Mr. James Dowe, Dikewood Corporation; for their achievement in producing the programs necessary for interfacing the remote terminals with the central processor.

The members of the staff of the Library of the Medical Sciences, University of New Mexico School of Medicine, should also be thanked for their helpful suggestions. Dr. Charles W. Sargent, Deputy Librarian, and Mrs. Virginia L. Algermissen, Associate Librarian, in particular, contributed substantially to the project.

We would also like to thank R. H. Fitz, M.D., former Dean; R. S. Stone, M.D., Dean of the School of Medicine; and other members of the faculty of the School of Medicine for their help.

The publication of this report was considerably simplified by the diligent efforts of Miss Shirley Laukonen, Technical Writer.

Robert T. Divett

W. Wayne Jones

TABLE OF CONTENTS

Part Page

I. General Systems Description

Introduction... 2

Systems Definition

The Batch System... 6
The Terminal System... 17
Equipment and Cost... 36
Recommendations... 39

II. Technical Descriptions

File Organization

File Structure... 48
File Construction... 49
File Manipulation... 50

Program Structure

Program Languages... 57
Resident Programs... 58
Transient Programs... 64

Bibliography... 70
Appendix A. Library Forms... 73
Appendix B. 2260 Screen Displays... 84
Appendix C. Source Listings... 97

PART I

GENERAL SYSTEMS DESCRIPTION

INTRODUCTION

Information in the medical field has been increasing at an almost exponential rate. Current statistics show that if the increase follows present trends, a large medical library (over 100,000 volumes) will double its size in less than two decades. A smaller one will double in less than one.[1,2] This veritable "explosion" of information has made librarians aware of the need for a revision in the system of handling information because present methods are slow and inefficient for such large volumes of work. In recent years, this had led to limited use of computers or other automated systems for easier processing of material. Some computer programs have provided easier search methods. However, the continued increase in amounts of literature and the demands for increased services to the user because of the added volumes of information, have shown that even these "batch" systems of processing, as these automated systems have been termed, are not sufficient. Instead there is a need for a total computer system which will economically and efficiently process and disseminate the literature in the library.

---

[1] Richard H. Orr and Alice A. Leeds, "Biomedical Literature: Volume, Growth, and Other Characteristics," Federation Proceedings, 23:1310-31, 1964.

[2] "Oregon's Computer Instruction Network," Educom, Bulletin of the Interuniversity Communications Council, 3:1-2, 1968.

Studies have shown that it is highly desirable and economical to automate the entire library by means of an "on-line" (or direct hook-up) time-shared computer system.[3] This gives the librarian continuous access to the computer for technical processing, circulation, and reference; and at the same time, the user has access to the files for searching. The high use factor in such a system (i.e. many tasks in one system performed at the same time) distributes the cost over a number of services, making it more economical to install the system for total use. Also, when a computer is on a time-shared system, the cost of operation is reduced, because the use of the computer and the expense are shared with the other departments serviced by it. These two factors make the total computer system highly economical.

The use of an "on-line" computer system will also increase the efficiency of library functions in the following ways: (1) Having library files in quick-access computer storage avoids duplication of effort in various library processes--order blanks, book cards, catalog file cards, etc. are typed once instead of separately for each process. (2) There is less chance of duplicating orders when total library holdings can be rapidly searched for a particular work. (3) Much of the element of human error is eliminated from library functions by use of an automated system. There is less chance for error when there is only one file to handle in all the phases of processing. (4) Searches are thorough and complete, especially if files have been increased by entries from major indexing journals. (5) The total time spent on any service of the library--technical, circulation, or reference--is greatly decreased. The overall result is a more efficient, economical system of library procedures and services.

---

[3] Audrey N. Grosch, "Implications of On-line Systems Techniques for a Decentralized Research System," College and Research Libraries, 30:112-118, March, 1969.

These concepts of economics and efficiency were the basis for the "Total System Computer Program for Medical Libraries" at the University of New Mexico Library of the Medical Sciences. Its objective was to develop an integrated computer program for the total operations of a medical library including acquisitions, cataloging, circulation, reference, a computer catalog, serials controls, and current awareness services.

Background

The Library of the Medical Sciences, a medium sized library (approximately 48,000 volumes) which serves as the library for the University of New Mexico School of Medicine, was ideal for this experimental computer program because much of what was needed both in files and in hardware was already in use or available. At the establishment of the library in 1963, R. T. Divett, Librarian, and R. H. Fitz, M.D., then dean of the School of Medicine, decided to process all library holdings in machine readable form. This was necessary for the "semi-automatic" unit record processing system first initiated and for eventual "on-line" operations. Consequently, all files were on punched cards which eliminated the usual lengthy process of establishing a machine readable file from which to work.

A keypunch, a sorter, a collator, and an IBM 407 accounting machine were in use when the grant was issued. These were used primarily in unit processes for cataloging,[4] circulation,[5] and printing of records. When the grant was issued, the system was changed into "batch" programs for use on an IBM 360 model 40 256K computer which was available through the Research Center at the University of New Mexico. The "batch" system was also expanded to include the actual processing of acquisitions, cataloging, and circulation. Once these were completed, work was begun on the total computer system which would add reference and search functions to these already performed by the "batch" program and combine them all in one computer program.

---

[4] Robert T. Divett, "Mechanization in a New Medical School Library I. Acquisitions and Cataloging," Bulletin of the Medical Library Association, 53:15-25, January, 1965. This article will provide more detailed information on the unit record procedure.

[5] Layde M. Payne, Louise Small, and Robert T. Divett, "Mechanization in a New Medical School Library II. Serials and Circulation," Bulletin of the Medical Library Association, 54:337-50, October, 1966. This article will also provide further information on the initial processing procedures.

SYSTEMS DEFINITION

To the reader, development of two nearly parallel systems may seem a duplication of effort, and in a sense, it is. However, when the project was begun, there was an immediate need in this particular library for a system which would perform most clerical functions. Time was an essential factor; and since the equipment for a total computer system was not available, the simpler "batch" programs were designed to meet the requirements.

Development of these "batch" programs proved a greater advantage than was anticipated. It produced a very workable system which is still in use in the library. Furthermore, working with library functions as units gave both programmers and librarians a better understanding of what could be done with a total computer system and how it would need to be done in the total system computer program which was to follow. Both of these systems are defined in the succeeding sections.

The "Batch" System

Until now quotation marks have always been used when speaking of the "batch" programs or system. This is because it cannot technically be termed a computer system but instead is a combination of manual and machine processing of the clerical functions of the library.

The clerical functions performed by this batch processing system are those in the acquisitions, cataloging, and circulation departments of the library. The system can also perform several specialized services. These functions and services, though very different from one another in purpose and process, are all tied together in one system by IBM punched cards which are established at the time of acquisition. Through these cards, a sequence of processing library materials has been formed. The best method of explaining and defining the batch system then is by following this sequence.

Acquisitions

When an item is ordered, a purchase request form is filled out either by the library user who is recommending the item for purchase or by the acquisitions staff member who is ordering the item. The form includes: author, title, place of publication, publisher, date, edition, volume, source of bibliographic citation, date ordered, and requester's name if he wants the item on advance loan when it arrives. (Figure 1, p. 74) This citation is checked by the acquisitions staff; and the vendor, estimated cost, accession number, copy number, and order number are assigned before the item is ordered. At this same time, a supplemental order form is manually typed and sent to the vendor as his order. (Figure 2, p. 74) The request form is then taken to the keypunch section where the abbreviated author, abbreviated title, copy number, accession number, vendor number, and order date are punched into the card. A code which indicates the item as a direct order, approval item, or gift is also punched into the card. This punched card is then used to create the Processing Information

List (PIL) Decklet,[6] which is a small deck of IBM punched cards containing the following:

Machine Card	(3-1)	
Payment Record	(3-2)	
Receipt Record	(3-3)	
Decision Record (optional)	(3-4)	
Cataloging Record	(3-5)	
Data Processing Record	(3-6)	
Preparation Record	(3-7)	
Filing Record	(3-8)	
Notification Record (optional)	(3-9)	(Figure 3, p. 75)

These are cards which have been pre-punched to perform their separate functions and are now duplicated with the information from the punched request form.

The Machine Card, which is kept in the keypunch section for further processing, is used to add the item to the Processing Information List. The PIL is a print-out from the magnetic tape file of all items in process from the initial ordering stage to final preparation for shelving. (Figure 4, p. 76) Besides abbreviated author by which the list is arranged, the list contains all the information which was punched into the Machine Card. Once the item is on the PIL, the remainder of the decklet and the request form are returned to aquisitions and filed until receipt or cancellation of the item.

When the item is received, the Payment Record and invoice are sent to purchasing for payment. The request form is then filed as a permanent record of the order. These are both manual processes which are now independent of the system.

---

[6] Marian R. Ausherman, William D. Wittekind, and Robert T. Divett, "PIL - A Control Procedure for Technical Services Management," Bulletin of the Medical Library Association, 55:394-8, October, 1967. See this article for further detail.

If the item as received is acceptable (i.e. it is the correct item and is not damaged), the Receipt Record is sent to keypunch where it is punched with receipt date. When run through the computer, this card then updates the PIL by adding that date to the tape file and by listing the date after the item on the print-out. (See Fig. 4) However, if the item is unacceptable and is not reordered or if the order has been cancelled, the "cancel box" on the Receipt Record is checked. That decision is punched into the card, and the entire record will automatically be dropped from the PIL when the list is updated. (The payment adjustment in such cases is still processed by hand.)

If the item ordered is initially an approval item, the optional Decision Record is inserted in the PIL Decklet. This card is omitted if the item is a direct order or a gift. After an approval item has been received, a decision must be made whether or not it is to be kept. If the item is rejected, a "no" decision punched into the card removes the entire record from the PIL.[7] If the item is retained, a "keep" decision is indicated on the Decision Record, and the card is sent to keypunch where it is punched with the decision date. When the PIL is updated, this date then appears beside the receipt date indicating that the item has passed the acquisitions stage of processing. The item and the remaining decklet are then sent to cataloging.

---

[7] Since the print-out of the PIL is of more concern than the tape file in this first section, it is hereafter assumed that "PIL" refers to this physical list. Of course, anything appearing on this list has been entered on the magnetic tape file. The technical descriptions appear in a later section.

Cataloging

Here the item is manually cataloged on a worksheet giving it full author, full title, imprint, collation, notes, tracings, and call number. (Fig. 5, p. 77) Once this process is completed, the Cataloging Record is keypunched with the date and is used to update the PIL by listing that date next to the decision date.

The item, worksheet, and remaining decklet are then sent to keypunch where cards are punched with all the information appearing on the worksheet. These are kept as a machine-readable file and are used to produce the conventional 3x5 catalog cards, the bookplate, and the pocket card, also known as the book card. After these printed forms have been produced and proofread, the Data Processing Record is dated; and the date is recorded on the PIL.

From data processing, the item, the decklet, the pocket card, bookplate, and catalog cards are sent to preparation. The bookplate and book pocket are affixed to the item; the pocket card is inserted. The hand-produced spine label is also affixed. Once this is completed, the Preparation Record is dated, and the date then appears on the PIL to indicate the item has completed the cataloging section of processing.

After all the above work has been checked and the catalog cards filed, the Filing Record is sent through data processing to remove the entire record from the PIL. The item is now ready for circulation.

At this point, all the cards used in processing have been taken from the decklet one-by-one at their appropriate stations and run through the computer to indicate on the PIL at what stage of processing the item is. The final Notification Record is all that remains. This is an optional card which is inserted in the decklet only if the requester wants the item on advance loan when it is through processing and ready for circulation. This card is mailed to the requester notifying him the desired item is available for his use. If there is no advance loan, the item is placed on the shelves for circulation.

From the above description it may sound as if the batch system is more of a computer system than it is. It may sound as though the cards are sent to the computer the moment a process is completed. This is erroneous. As each segment of the processing is finished, the cards are punched with that date; but the cards then cumulate with those from other stages of processing until they are all run through the computer together. In this library, the cards which update the PIL are run once a week.

Knowing that the bulk of the processing of acquisitions and cataloging is done manually in the PIL program and knowing from the above that the PIL is a limited computer program, the reader may wonder what advantages the system has over the regular manual system.

One of the greatest advantages of this system is the elimination of duplicate typing of bibliographic information. The information is given once on the request form, typed once on the supplemental order, and punched into IBM cards. After this is done, the typewriter is not used again until the final preparation stage when the spine label is manually typed. Of course, the keypunch is used during each process, but this use is minimal except in the initial ordering and in the cataloging stages. Typing is also eliminated from the preparation of catalog cards, bookplates, and pocket cards which are all produced from IBM cards.

Because of this elimination of duplicate typing and duplicate handling of the same information, less time is spent in processing; and items are available for patron use more quickly than in a manual system. This is advantageous both to the user and to the library staff.

The PIL, with its physical list of all items in processing, has distinct advantages over a manual system. Items can be quickly traced and located--merely looking at the list reveals in which stage of processing an item is. Consequently, the location of new library holdings is always known should they be needed for either patron or staff use. Duplication of orders is also less likely to occur when a listing of all items in processing can be checked before ordering.

Listing the processes by date lends itself to another use of the PIL, but this involves one of the special programs previously mentioned. A program which will analyze the contents of the PIL has been written as a separate batch program. Given the proper information, the program will instruct the computer to search the entire list and print out any item which has been in any specified station since a specific date. The necessary information (the station and the date) is fed into the program by punched cards, and all items meeting the stipulations of the program will be printed out by date, beginning with the date given. Thus if a library staff member wants to take action on all items ordered since month/day/year which have not been received, he specifies the order file and the selected date and receives a list of all items which meet these conditions. He can then act on them as he desires.

Another special program used in the system will produce a monograph shelf list, or sequential call number listing of all monograph holdings in the library. (Fig. 6, p. 78) This list is printed from its magnetic tape file approximately once every two months giving catalogers a list to use to ensure that the same call number is not given two separate monographs. The file, which is created from the punched shelf list cards produced during cataloging, is updated when sufficient data is available to warrant running the entire list.

A program similar to the one which produces the monograph shelf list is used to produce a serials shelf list. (Fig. 7, p. 78) Like the monograph shelf list, the serials shelf list is based on IBM cards which are punched during cataloging; but unlike the monograph shelf list card which includes complete bibliographic information, the serials shelf list card contains only title, accession number, vendor code, frequency code, and ownership code.

After a serial has been received and manually checked in, the shelf list card is taken to the keypunch section where an issue shelf list card is made by duplicating the original card and adding to it volume number, issue number, and date received. This card in turn produces the pocket card which is placed in the issue. The issue shelf list card then goes to the computer to place that issue on the serials shelf list which is an alphabetical listing of all serial holdings.

The serials shelf list is printed once a month but is supplemented daily by a list which cumulates each day until the end of the month when the two lists are merged. This list is useful to the patron in locating references and to the librarian as a complete up-to-date list of holdings.

Circulation[8]

Thus far, this description has been limited to the acquisition, cataloging, and related segments of the batch system. The details of the circulation segment have not been given because the circulation batch is a separate unit. It is tied to the other segments by the pocket card which has been produced during cataloging; but beyond that, it is a separate process which has its own sequence.

The process begins after a patron has selected an item which he wants to charge-out of the library. He takes the item to the charge-out desk where the pocket card and borrower's identification badge are presented to the library staff member. (Fig. 8, p. 79) (The identification badge is a plastic card which has been punched with the borrower's number-a number unique to each borrower but which indicates his status as doctor, medical student, nursing student, staff, etc.) The two cards are inserted into an IBM 357 which produces two other cards on a slave keypunch in the 357 system. These cards contain the duplicated information from both inserted cards--the bibliographic information and type of loan (either book or journal) from the pocket card and the borrowr's number from the badge--plus the date of loan and period of loan which the 357 system adds. One of these cards is placed in the book pocket with the original pocket card; the other is sent through the computer to put the item on the circulation list. To prevent confusion in identifying these cards from each other and from the pocket card, these cards are different colors.

---

[8] Payne, op. cit., pp.337-50. See this article for further information.

At the end of each day, the circulation cards are run through the computer to add these items to the circulation list. (Fig. 9, p.80) The list is divided into sections by type of item. The listing within each section is alphabetical by main entry and also includes title, call number, accession number, borrower's number, and date of loan. The charge-out procedure is complete once the item is on the list.

When an item is returned, the return card is taken from the pocket; and the item is reshelved. The return cards cumulate each day and are then run through the computer to remove the circulation record from the list.

Once a week a list of overdue items is produced by combining the circulation list in its tape form with the patron tape, which is a list of the names and addresses of users by borrower's number. (This list is flexible and any information can be added or deleted at any time.) The convergence of these two lists produces printed overdue forms. (Fig. 10, p.81) Each form includes all the overdue items charged out to one patron and are mailed to the user in "window" envelopes.

The above function of the batch system concludes the usual circulation procedures of the library, but the batch system goes one step beyond the usual manual procedures. Statistics in a manual system are usually compiled from a physical count of cards taken before the cards are filed each day. This limits the type and amount of statistics which can economically be taken. A special program in the batch system has eliminated the usual restrictions of the manual system. Since the establishment of the library, a circulation statistics tape has been kept on every item circulated. The program can analyze this tape file and produce statistics on the type of borrower, the type of item loaned. It can also limit these statistics to any length of time down to monthly periods. These detailed statistics are much more useful and accurate than those compiled in the manual system.

As a whole, the batch process of circulation is faster than the manual system for both patron and staff. There are no charge-out cards to fill out, and all filing is done mechanically--just two examples of the reduction in time. The process is more nearly accurate--the computer does not overlook a card or a tape record. Printed lists are much more convenient than the boxes of 3x5 cards which characterize the manual system. These are just a few of the advantages of the batch process of circulation.

The entire batch system has many advantages over a manual system. Most of them have been given as the processes have been described, but there are two which characterize the whole system and which make the design and operation of the system worthwhile. These are the increased speed and greater accuracy in all phases of processing and handling. Thus, the result of the batch system is an efficient means of handling and disseminating library materials.

The Terminal System

Though the batch system may be an efficient means of handling all processing phases of library materials, it leaves much to be desired in the total operations of a library. It has not touched upon the reference section, nor has it really effectively minimized handling materials in the processing section--files and materials are still handled in segments, and each phase requires handling the same materials over again.

If a library is to be totally mechanized, these problems must be overcome; and the only effective means of overcoming them is with an "on-line" computer system with "terminals," or direct access computer units, in each section of the library, including reference services and patron use.

The terminal system, as this total computer system is called, is a direct access system. The terminals to the IBM model 40 256K computer are IBM 2260 display stations[9] which consist of a typewriter keyboard and a viewing screen. The programs in the system have been written so all communication between the user of the 2260 and the computer is done in a series of questions or commands which appear on the screen. All the user has to do is type in his answer or type in the instruction from the command, and the computer will act on the information it has been given. (This is commonly termed the "conversational mode.") This concept will be better explained by example as the system is defined. For ease in understanding and for easier comparison to the batch system, the terminal system will be explained by describing the sequence which is formed when an item goes from one processing phase to another throughout the system.

---

[9] The IBM 2260 display station will hereafter be referred to as the 2260.

Acquisitions

The acquisitions process begins when a request for an item comes into the library or when an order is approved. This information is given to an acquisitions staff member who begins the computer process by first typing his identification number[10] and the word "order" on the 2260 to identify who the user is and to indicate which file he wants to enter. (2260--1-2)[11] If he is not qualified to use this phase of handling, his number will not match those on file for users of this process; and he will be instructed that he cannot use the equipment. (2260--3) If the number does match, the system will respond with a command and the processing can begin.

---

[10] The identification number is a unique number given to each user of the library. It indicates his status as doctor, medical student, nursing student, library staff, etc. and identifies the person by number, name, and address.

[11] The (2260--) notation throughout the text refers to the information which is displayed on the screen of the 2260. The number following the dash refers to the illustration numbers in Appendix B where there is an example of each screen display.

When an item is ordered, a skeleton file of the bibliographic and administrative information of the item is created in computer storage by typing the data on the 2260. (2260--4-11) The bibliographic segment of the file is a partial file of what information is available on the item and includes author, title, imprint, and the assigned accession number, which includes the copy number and volume number if applicable. The administrative segment is the information used in accounting and includes vendor number, estimated cost, purchase order number, and order date. From this information, a purchase order form is automatically printed by the computer system. (Fig. 2, p. 74) The vendor number gives access to the name and address of the vendor which are printed on the form so it can be inserted into a window envelope and sent directly to the vendor. The form also includes author, title, accession number, and order date.

Thus far in the sequence, time and effort have been saved--even in comparison to the batch system. All the basic files and the purchase order for the item have been created in one typing rather than the usual three or four of the other systems. Even the batch system requires two such operations.

Duplication of orders is also easier to prevent because all existing orders and all the library holdings are in storage together and are automatically searched by the computer as the item is entered so a duplicate cannot be ordered inadvertently. This is a precaution which is built into the system in case a duplication of order is not caught in the first stage when the acquisitions librarian uses the search process (which is described later) to completely examine the files. The librarian first searches by every possible author and title entry before an order entry is made. The above automatic process is just another check of the files.

When an item is received, the request form or duplicate copy of the order form is examined to determine the accession number which the item has been assigned. This number is used as the access point into the files. The user then types his identification number and the file word "receive" into the 2260. (2260--12-13) If he is a qualified user of this process, the system will respond and direct the user to enter the accession number of the item. The user is then asked if he wishes to cancel or retain the item. If he cancels it, the skeleton file is automatically dropped from storage. If the item is retained, the user is asked to update the administrative segment of the file with the date received, the actual cost, and the invoice number. (2260--14) The acquisitions department then passes the item with its accession number to cataloging. (2260-15)

A special process which has not been developed by this research but of which the system is capable is an automatic search of outstanding orders (items which are on order and have not been received). The average period of time from order to receipt can be calculated from the respective order and receipt dates, and action can be taken on any item which is long overdue its expected date of arrival. A form which can be mailed directly to the vendor can be printed by the system, or a list similar to that in the batch system can be printed and then manually examined to determine the course of action. This process would not necessarily be in the regular sequence of processing but would be run periodically to examine the order files.

As in the batch system, the payment record for an item can be initiated by the computer in the terminal system. After receipt of the item, a form which includes the vendor's name and address, the actual price, the title, and the author of the item can be printed and manually sent to the voucher clerk for payment. If the business office of the library is "on-line" with other payment functions of the institution to which the library belongs, the system could be designed to automatically store this information in the payment segment of the files. In either case, the payment procedure is then no longer a library function.

Cataloging

The cataloging process is begun when a cataloger enters his identification number and the file word "catalog" on the 2260. (2260--16-17) If he is an approved user of this phase, he will then be asked to enter the accession number of the item. (2260--18) When this number is typed in, the entire entry for the item appears on the screen and any data except the accession number[12] can be corrected. The cataloger can then finish cataloging the item by changing the author and title entries if they were not correctly given in acquisitions and by adding imprint, tracings, notes, and collation. (2260--24-41) To change either the author or title entries in the file, the entire author or title record must be dropped and the new information is then typed in. (2260--18-23) Any other part of the entry (except the accession number) does not need to be dropped to be corrected. The system merely replaces the incorrect data with correct information. Any added information is just typed in on command from the system. (2260--24-41)

---

[12] The accession number is permanent in the files. To change it the whole entry must be dropped and a new entry made.

The number of tracings which can be added--whether author, title, or subject--is unlimited. (2260--19, 23, 25) The number of subject tracings is actually expanded automatically in the system by the National Library of Medicine Medical Subject Headings (MeSH) list which has been incorporated. (Fig. 11, p. 82) The MeSH list is a list of subject headings which has been approved by the National Library of Medicine and which are arranged into a tree structure so that from any selected subject heading a more general or more specific topic can be found.

It is assumed that the subject tracings given during the cataloging are the most specific for the item; but in searching, it is advantagous to have reference to other subject levels. This is the reason "see" and "see also" references are given, but these usual references are not practical for a total computer system. Furthermore, with the MeSH list they are not needed. Instead the tree structure is programmed into the system so that when an item is cataloged at any given level, the more general subject headings are automatically given; and the item can be searched through the more general topics as well as the specific. Thus if an item was given the specific level subject tracing "Pediatrics," the reference would automatically be listed under "Specialties, medical" and "Health occupations"--the more general subject headings above "Pediatrics." (The necessity for this automatic expansion of subject headings will be more clearly understood after reading the Searching section of the terminal system.)

During cataloging, a subject which is not a National Library of Medicine heading may be entered. The system will indicate on the screen that the subject heading is not in the files, and the cataloger can either change the subject tracing to conform to the MeSH list, or he can leave the entry as it is. Of course, if an entry which does not comply with the approved subject headings is made, the tree structure is of no advantage.

The cross references for title and author entries are not automatically built into the system as are the subject references. The author and title cross references must be added to the system by the cataloger. After they are added, the system will automatically go from one reference to another. The terminal user does not have to re-enter the system with a new entry. Thus, if Theodore Henrijk van de Velde's _Ideal Marriage, Its Psychology and Technique_ were cataloged, the cataloger would include an author cross reference from van de Velde, Theodore Henrijk, to Velde, Theodore Henrijk van de, and a title cross reference from the abbreviated title, _Ideal Marriage_, to the full title. If the user tried to search for the item under Van de Velde, Theodore Henrijk, or under the abbreviated title, _Ideal Marriage_, the system would accept his entry, find the correct entry by the cross reference, and display the correct full citation on the screen. If, however, these cross references had not been added by the cataloger, the system would have given a negative answer to the request.

Once the full author, full title, imprint, and tracings are in correct form, the remaining cataloging information is added to the entry in a series of questions from the system. (2260-- 33-41) This information includes the call number; copyright date; publication date; the date cataloging is completed; the type of main entry--whether individual author, corporate author, title, etc.; a notation which indicates the item is the result of a conference, meeting, symposium, etc.; the language of the original work; two languages of translation; the date an item was reproduced if it is a reproduction of an original; the National Bibliographic number--a number which is not in use at this time; the National Library of Medicine MeSH number--the separate number given to each subject heading in the MeSH list; notes; analytics; and collation. All this information is merely typed into the system on the 2260, and any part of it can be change at any time.

It is not necessary to include all the above information for each entry in the files, but the cataloging programs have been written to include them all, should they be needed to completely identify an item. The only necessary information in the citation is that which is needed by the user and staff to search and locate an item in the holdings. This includes the call number and the information which is used as a search method or used to delineate a search while in progress. (See Searching section)

Entrance of the collation is a signal to the system that cataloging is complete, and the system can begin printing the book plate, spine label, and pocket card for the item and the 3x5 catalog cards as the back-up file for the computer.

Once these items are printed and either affixed to the item or filed, the item is ready to be shelved for circulation.

Searching

One of the greatest advantages of a total computer system is the mechanized or automated search. The user no longer has to plough through hundreds of 3x5 cards, but can sit at a terminal and have most of his work done for him at a faster rate than he could do it himself.

To begin a search, a user enters his identification number on the terminal. (2260--43) As in the other processes, this number is matched by the computer with a list of eligible users of this phase of the system. If the number does not match a number in the list, the system indicates that he is not a qualified user and will not accept any information from him. If the number is verified by the computer, the search procedure begins.

The user is first asked to define the methods of his search. (2260--44) He is asked to search either by author, title, subject, or any combination of these choices. He can also choose a logical search, as a subject-and-subject or author-and-author search is called.

Once the user has made his choice and specified it on the 2260, (2260--44) he is asked to type his search entries on the terminal. (2260--45) The computer system then searches the holdings file and replies with the total number of entries which meet the requirements of his search. (2260--46)

If there are more than three entries, the user is given the opportunity to redefine his search by delineating the language of his citations, by limiting his items to either books or journals (the system will later include journal articles), or by specifying the dates of publication to which he wants his citations limited. (2260--46) These three options are immediately given in an author or title search with over three citations. However, if the search is by subject, the user is first asked to redefine his search by narrowing his subject; and he is given the possible choices of more specific elements from which he can choose. (2260--54-55) When the subject search is at its most specific level, the user can then redefine his search by the above three options. Once one of these choices has been made, the citations which meet the stipulations are displayed on the screen. (2260--56)

If there are less than three citations under any entry of the search, they are all automatically displayed on the screen of the terminal and the search is concluded. (2260--47)

There may be no citations listed in the files for a particular entry. In such a case, an author or title search is ended. However, if the search is by subject, the system will suggest a broader topic under which to search to find information. (2260--61-62) The user can then begin a new search under this subject.

A search does not have to be brought to its lowest, or most specific point before the citations will be displayed. By refusing to redefine the search at any point along the way, (2260--62-63) the user can have the entire list of citations displayed. However, if the list is lengthy, this becomes cumbersome to the user.

If the item for which a user is searching is on order or in processing at the time of request, the system will indicate this status to the user. Of course, this notation will only appear on an author or title search. (Since an item is not given subject tracings until one of the last phases of processing, it is almost available for use once the subject entries have been listed. Thus the "in processing" note is of no real value in the subject search.)

Besides quickly searching the entire files of the library, the search process has one other feature which saves time for the user. Once the needed citations have been found, the user can depress a "print" button on the 2260; and the citation on the screen will be printed by the computer system. The user can then go to the printer and obtain a copy of his citation; he does not have to take the time to copy it down.

In addition to the precaution taken by having each user of the system identified and verified in a user list, the system has another built-in safe guard. Only the terminals which are for staff use can be used to process items or to change the original files. The terminals which are for patron use are restricted to the search funtion. Thus, the files cannot be accidently destroyed by the users of the system. This also ensures that terminals will be available at all times for both staff and patron use.

Circulation[13]

The circulation procedure of the terminal system is not a completely separate unit as it was in the batch system. It ties in with the other processes not only by the pocket card which is produced during cataloging, but also by the "on-line" computer files which the processes share.

The circulation procedure begins when a patron has searched and located an item he wants. He takes it to the circulation desk where he presents the item and his borrower's identification badge (Fig. 8,p.79) to the staff. The pocket card from the item and the borrower's identification badge are inserted into an IBM 1030 system which puts the borrower's number and date of loan directly into the description section of the entry on computer file. When another user searches for this item, the borrower's number and date of loan then appear on the screen of the 2260 as part of the bibliographic citation to indicate that the item is on loan and not currently in the collection. This saves the user the time and trouble of looking for an item which is out of the library.

At the same time that the descriptive file of library holdings is updated with the borrower's number and the date, a second file is created. This is the circulation file of all items out on loan and includes the borrower's number and accession number of each item. (The use of this file will be explained below.)

---

[13] Due to the lack of funds needed to acquire the equipment which would be necessary to maintain the circulation files, the research of the circulation process of the terminal system was not completed. However, it would not take long to complete the entire procedure if the funds were available. Consequently, the entire section is written as if the process were in use.

At the time of charge-out, the terminal system also produces a date-due slip which is included with the item when it leaves the library. This is just one possible method of indicating to the user when the item is due. Any number of manual or machine procedures could be used instead of this particular method.

The terminal system also has another charge-out procedure. With the "on-line" computer system, it is possible to install IBM 2260 units in areas other than the main library building. The user searches from these remote terminals following the regular procedures. Having decided on the necessary item, he then depresses the "print" button on the terminal which activates the printer in the library. A student aid on duty at the printer notes the source of the request from the terminal number which appears on the print out. He locates the item, charges it out to the borrower's number which also appears on the print out, and delivers the item to the user. (The IBM 1030 system has a manual input so an item can be charged out without the use of the borrower's identification badge.) The remote terminals and this alternate charge-out procedure make the information in the library more easily accessible to the user and expand the services the library provides.

The check-in procedure of the terminal system begins when an item is returned to the library. The pocket card is removed from the item and reinserted in the 1030 system along with a check-in badge which indicates to the system that the entry is to be removed from the circulation file and that the borrower's number and date of loan are to be removed from the holdings file. The item is then ready for shelving once the pocket card is replaced in the item.

Both of these procedures--the charge-out and the check-in--have an advantage over other computer or manual systems. They are up-to-date the moment an item is entered in or removed from the file. There is no period of lag between the time an item is charged-out or checked-in and the time it appears in the circulation files as there are in other systems. The terminal circulation process also requires less handling. There are no punched cards to cumulate and run each day. There is only one processing step in each of the charge-out and check-in processes. As a whole, the terminal circulation process is more efficient and more smoothly handled than either the batch or the manual system.

The circulation procedure includes more than charge-out--check-in processes. It also includes inquiries and overdues. When a patron of the library cannot find an item which he wants and a search on the 2260 shows that the item is in the collection, the patron checks circulation to see if it can be found. The circulation staff member types in his identification number and the keyword "inquire" on a 2260 at the circulation desk. If the number is verified, the computer will ask if the inquiry is of a book or a patron. Since the user is checking the circulation file for an item, he replies that it is a book inquiry. He is then requested to enter the title of the item for which he is looking. The computer then searches the circulation file for the item. If the item is in circulation, and the notation of this fact has not been made on the holdings file, the computer system will reply that the item is on loan and due on the listed date. The reply will also include the borrower's number which can be used to identify the borrower if it is necessary to locate the item. If the item is not in the circulation file, the system will also indicate that there is no record of the item in circulation, and other methods of locating the item must then be employed.

Another form of inquiry which the circulation procedure includes is the patron inquiry. This inquiry is used to obtain a list of all items charged out to a particular user. The staff member again types in his own identification number and the keyword "inquire" on the 2260 at the circulation desk. The number is verified, and in response to the computer's question whether this is a book or patron inquiry, the terminal user types in "patron inquiry." The computer then asks for the patron's identification number. Once this is typed in, the computer searches the circulation file by borrower's number and prints out all the items which are charged-out to this number. The list includes the bibliographic information and the dates the items are due. This part of the circulation process is especially useful when a patron is leaving the staff or when a patron is leaving at the end of an academic term, and the library staff wants a complete list of all the items he still has on loan.

The circulation procedure also includes the processing of overdue items. Each day before the system is brought up for use (changing the date, etc.), the overdue program is run. The terminal user initiates the program by typing in his identification number and the word "overdue." The computer verifies the number and instructs the system to search the circulation file for any items which are overdue and to print the overdue items on forms which include the name and address of the borrower, a list of the overdue items, and the dates on which they were due. These forms are printed so they can be inserted into envelopes and mailed directly to the borrower. (Fig. 10, p. 81) Duplicate copies of the forms are kept as library records.

This daily checking of overdues is an advantage over the less frequent compilation of other systems because it notifies the patron that the item he has on loan is overdue as soon as it becomes overdue. There is also a greater chance that an overdue item will be returned more quickly if the borrower is reminded daily that he still has an item which another patron may need.

This aspect of the circulation process is particularly useful in a system where there are no fines. If a fine system is part of the library policy, the computer system can include an accounting program which will automatically tabulate the fines. This amount would then appear on the overdue forms which are sent to the patron.

Besides the daily checking of overdue items, the circulation procedure contains another program in the overdue section of the process. If a borrower has items charged-out which are overdue, and he wishes to take out any other items, the computer can refuse to charge-out any items on his borrower's number until the overdue items are returned. This program is a safeguard and a control over patron use, but it can be overridden should a patron really need an item despite his overdue record. This aspect of the program is essential, especially in a technical library where materials are needed for research.

Like the batch system, the terminal system also has a special program which is used to produce circulation statistics. In fact, the statistics are obtained in the same way in both systems. In the terminal system, whenever an item is charged out of the library, the bibliographic information and borrower's number are stored on a tape file as well as in the circulation file. As in the batch system where circulation information is also stored on tape file, the terminal system produces circulation statistics from this tape. The statistics obtained include the type of patron, type of loan, and total number of items loaned. The statistics can be narrowed to monthly or even smaller segments of time.

This program of the terminal system is not "on-line" with the computer, but instead is a batch-type of program. Since it is run so infrequently, it is not practical to put it "on-line" with the system; the tape storage and batch running for this particular process is much more economical and practical than an "on-line" process.

The terminal system does not have a program which will produce either a monograph shelf list or a serials shelf list, but in the terminal system these programs are not needed. Since catalogers can use the terminal and the holdings file to check for duplicate call numbers, there is no need for a printed monograph shelf list. The patron and serials staff both use the terminal to check the serials holdings; the staff also uses the 2260 to check-in new issues as they come into the library. Consequently, the serials shelf list is not needed. However, if it were found that these lists were needed for library operations, the programs which would produce them could easily be written.

The above description of the circulation process concludes the definition and description of the terminal system, which thus far has been shown as a total computer system for all phases of library processing and material handling. A total system can even go beyond the acquisitions, cataloging, searching, and circulation processes and the special programs which have been described in the preceeding pages; but time and money have not been sufficient in this project to completely examine the capabilities of the system. Further possibilities and uses for the total computer system are described in the recommendations section of the manual.

The system, as it is, is a total computer system for all library information processing and has many advantages over the batch system and over the manual systems. One of the greatest advantages is the speed at which all processing phases are done. The time spent in any particular handling phase has been reduced--especially in the user searching process. Duplication of effort has been eliminated; consequently, the system is more efficient--less time is spent doing the same processing. Manual handling has also been cut to a minimum so there is less chance for human error than in a system which depends heavily on manual processing. Another advantage is that the location of any item is known at any time through all phases of processing from ordering to circulation. Thus a complete search for an item is made by merely checking one source--the holdings file which is accessible through the 2260. As a whole, the terminal system is a faster, more efficient method of handling all phases of library information processing than either the batch or manual system. It is also the only effective means of processing large volumes of library materials and of providing the user the services such large volumes of material require. Thus, the terminal total computer system is the only effective, efficient way of meeting the demands of service and processing that are required by a technical library.

An explanation of the equipment and the total cost of both the batch and terminal systems is given in the following section.

Equipment and Cost

The equipment and cost requirements of both systems are listed in this section in tabular form. The figures in the cost column are approximate figures and are based on the rate per month for leased equipment.[14] The overall cost and other requirements are summarized at the end of the section.

The Batch System

Equipment	Cost
Basic configuration:	
IBM 360 Model 40 256K Computer	$100.00 per hour[14]
including: IBM 2540 Card Reader	
IBM 1403 Printer	
IBM 2400 Tape Storage Drive	
IBM 357 Data Collection System	
IBM 357 Input Station	$ 47.00 per month
IBM 358 Input Station Control	$ 79.00 per month
IBM 24/26 Output Station	$100.00 per month
IBM 361 Read-Out Clock	$ 39.00 per month
IBM 360 Clock Read-Out Control	$ 50.00 per month
IBM 026 Printing Card Punch	$ 63.00 per month
IBM 029 Printing Card Punch	$104.00 per month
IBM 056 Card Verifier	$ 72.00 per month
IBM 082 Card Sorting Machine	$ 90.00 per month

---

[14] Since the cost of the computer varies so greatly and depends entirely on the length of time the computer is used, the leased price is given as rate per hour.

The Terminal System

Equipment	Cost
Basic configuration:	
IBM 360 Model 30 64K Computer including: IBM 2400 Tape Storage Drive	$100.00 per hour
IBM 2701 Data Adapter Unit	$295.00 to $795.00 per month depending on total configuration
IBM 1050 Data Communication System	
IBM 1051 Control Unit	$ 140.00 per month
IBM 1052 Printer Keyboard	$ 65.00 per month
IBM 1058 Printing Card Punch	$ 140.00 per month
IBM 2848 Display Control and Non-Destructive Cursor	$ 410.00 per month
Expansion Unit	$ 57.00 per month
Expansion Unit	$ 46.00 per month
Display Adapter	$ 656.00 per month
Non-Destructive Cursor Adapter	$ 40.00 per month
IBM 2260 Display Station	$ 816.00 per month for 16 displays
IBM 1053 Printer	$ 50.00 per month
IBM 1030 Data Collection System	$ 90.00 per month
IBM 1031 Input Station	
IBM 2321 Data Cell Drive	$2885.00 per month
IBM 2841 Control Unit	$ 720.00 per month
IBM 3021 Data Cells	$ 250.00 per month for 10 cells
IBM 2311 Disk Storage Drive	$1180.00 per month
IBM 1316 Disk Pack	$ 30.00 per month
Bell System	
Leased Telephone Lines	$ 4.00 per quarter mile
4B Conditioning and Balancing	
First Station	$ 37.50 per month
Additional Stations	$ 20.00 per month
201-B2 Data Sets	$ 140.00 per month for 2 data sets

Based on the above figures, the total estimated cost of the two systems is $644.00 per month plus the cost of the computer for the batch system and $8171.50 to $8671.50 per month plus the cost of the computer for the terminal system. This is for the equipment alone. When examining the overall costs of these two systems, the personnel requirements of the computer systems must also be considered. In addition to the usual library staff, the batch system requires a computer programmer and two keypunch operators. The terminal system requires a computer programmer. The salaries of these additional staff member must be included in the cost of the systems. However, these salary figures vary too greatly to give an estimate of these costs. These figures can be obtained when each system is considered for installation. At that time, the equipment costs should also be re-examined since they too will vary slightly.

This section concludes the definition of the two systems. The recommendation section which follows will describe further capabilities of the terminal system.

RECOMMENDATIONS

A total computer system is not strictly limited to the processing and handling of library materials. It has greater capabilities than these alone. Certainly, the primary aim of the terminal system is to minimize effort and to computerize the technical processing and patron searching of library materials; but the system can be adapted to include aspects which would increase the services available to the user and which would also increase the area over which material could be available.

One aspect of the system which would increase user services and increase the scope is sharing the system with several other medical libraries. The terminal system has been designed to be utilized in this manner. Several libraries linked to the same basic computer configuration can share the entire system. Their holdings can be placed on file together, and a search through a terminal examines the holdings of all the libraries. Obtaining an item which a library in the system may have becomes a simple interlibrary loan rather than a more complicated search for the location of the item through the usual interlibrary loan methods. The computerized search is also thorough and up-to-date; therefore, the item need not be loaned from a more remote location or may not need to be ordered for the collection when it is available from a cooperating library.

When one of the libraries in the system is ordering an item, the acquisitions librarian can search the holdings of all the libraries to determine if the item is in a cooperating library's collection or "on order" by another library. Duplication of items in the collections of the various libraries can thus be reduced so a greater diversity of items can be purchased for the total holdings of all the libraries. If, after checking the holdings, one library decides to order an item which another library has in its collection, the expense of cataloging the item is saved. The catalog entry is already on file; all that is added to the entry is the new location of the item.

Sharing the system greatly increases the use of any one library collection because the information in one library is not only used by the patrons of that library, but it is also available for use by the patrons of the cooperating libraries.

Sharing the system with other libraries also increases the efficiency and reduces the cost of the system. The files of the terminal system are designed so that the more information there is on file, the more efficient the storage. All similar entries are stored in one record. (i.e. All authors whose last name is Clark are stored in one file record.) The greater the amount of similar information on file (i.e. the more Clark's on file), the more entries there are in one record. The same amount of space holds more data; therefore, the storage is more efficient. This file structure makes sharing the system advantageous--several libraries will have more similar information to store than a single library.

The cost for each library in the system is reduced if the system is shared. If several libraries share the files and the basic equipment, the cost of the system is distributed among them; thus the cost is less for each library than if one library were responsible for the entire system. Considering this fact and the above discussion of the shared system, it is highly recommended that the total terminal system be shared by several libraries.

Because of its flexibility, the system could also be used and shared by non-medical libraries, but a change in the subject structure of non-medical terms would need to be made to make the system as effective as it is for the medical fields. This is an external change, not a change in programming technique or in the system. To use the file structure to its greatest advantage, a tree structure similar to the MeSH list would need to be built for non-medical subjects. Non-medical tree structure terms can now be entered in the files, but all subject tracings need to be added by the cataloger. However, a user can only enter and retrieve his specific subject; he cannot automatically move within the subject structure to more specific or more general subjects. If a tree structure for non-medical terms were developed and stored in the computer, the system would then have the same advantages for other libraries as it does for medical ones.

Another aspect of which the system is capable is using part of the storage files for uses other than those of processing and handling library information. A terminal could be available for the use of the administrators of the library or of its parent institution. Budget figures, statistics, etc. could be entered and retrieved by the system. In fact, the system has been designed to handle almost any type of data base. Thus, part of the files could be used for medical records, part for the teaching staff of the medical school, another part for tumor registry, etc.--the uses of the files of the system are almost unlimited. In each of these cases the files could be entered only by qualified users of each part of the files, and the terminals themselves could be restricted to enter only specific files. Therefore, the whole system could be used for any number of purposes without danger of unqualified users entering restricted files. This aspect makes the system flexible and applicable to uses other than those of the library alone.

As this discussion has shown, the terminal system can be used to perform functions which are not strictly library functions. It can also further improve those operations which are part of library processing. The terminal system has been designed to include a total serials control program. The research for the serials control has not been completed, but a program which would completely control the receiving, claiming, and processing of serials could be written. The monumental task which the handling of serials presents would be cut to a minimum by putting serials control "on-line" with the other phases of processing.

With an "on-line" system the actual check-in of serials could be done through a terminal. On receipt of the item, the librarian would update the holdings file of that serial with the volume number, issue number, and date received. Once this information was entered, the system could automatically produce the pocket card for the serial, and the item would be ready for circulation after the book pocket and pocket card were placed in the item.

Using the receipt dates which are entered into the serials files, an automatic claiming program could be developed. The system could examine the average length of time between issues of a specific serial; and if an issue were long overdue its expected date of arrival, the system could bring this fact to the attention of the serials librarian. The librarian could then evaluate the situation and decide if a claim should be made. If a claim were made, the librarian would enter the decision on the terminal, and the system would record the decision, record the date the claim was made, and print out the claim which would then be sent to the vendor or publisher. With this process the clerical functions of the claiming procedure are eliminated, yet the decision is left to the librarian.

A similar man-machine operation for serials is the binding procedure. A predetermined code based on frequency of the serial, on use of the serial, and on the volume and issue numbering sequence could be programmed into the system. Using this code, the system could indicate to the librarian that a specific serial should be sent to the bindery. The librarian could evaluate the notation and make the decision. If the decision to bind the serial were made, the librarian would enter this decision on the terminal. The system would charge-out the volume and issues to the bindery and then print the binding slips which would be sent to the bindery with the item. The clerical functions are again performed by the computer system.

The system has also been designed to further control serials by transferring the journal indexing information from the National Library of Medicine MEDLARS tapes to computer storage. No programming has been devoted to transferring this information, but the program which would examine the tapes and store all the information about the journals which the library holds could be written. Complete MEDLARS tapes go back to 1964. Thus, if the indexing information were on file, a search of the computer files would be complete for all library holdings from 1964 to the present.

The system has also been designed to accomodate the MARC tapes. The cataloging information from the MARC tapes could be transferred directly into computer storage, thus saving valuable cataloging time. The programs which would transfer this information have not been written though the system has been designed to accept such information.

Including a video tape file in the terminal system, would greatly increase the distance over which library information could be used. The actual text of monographs and serials could be stored in the video file. Remote terminals could be connected to the system; and through a console similar to the IBM 2260, a user could search the computer files, select an item, and have the actual text displayed on the screen of the console. A facsimile printer connected to the console could be triggered by a button on the terminal, and the information which is displayed could be reproduced by the printer. This application of the terminal system would require more equipment than the system now uses. Copyright laws need definition or amendment regarding this type of use. However, once these problems were solved, the system would make the information on file at one location available for use at other locations. It may also be possible to reduce the storage space required in the base library or location if the video file were used.

In the terminal system there are several equipment substitutions which can be made. Instead of using an IBM 2260 with the IBM 2848 Display Control, an IBM 2265/2825 Display System can be used. The IBM 2265/2845 is an independent system which eliminates the cumbersome IBM 2848 and reduces the cost if just one or two terminals are needed. The IBM 2265 is $205.00 per month leased and the IBM 2845 (the control unit for the IBM 2265) is $175.00 per month leased. A teletype may also replace the IBM 2260 and IBM 2848, but its slower speed reduces the capabilities of the system. The cost is $62.00 per month for the basic unit.

These substitutions for the IBM 2260 and IBM 2848 would be less expensive. However, before they are implemented, their capabilities would need to be compared. It must also be considered that the terminal system as now written will not accept data from either of these input-output components. The programs would need to be modified if they were used. These components would be cheaper for one or two remote consoles, but for a larger operation the IBM 2260/IBM 2848 is more economical.

If all these recommended uses were incorporated, the terminal system would be a totally automated library system. It would be available for library and non-library information users and could be used and shared by other libraries or terminals.

In the foregoing sections, the batch and terminal systems and the systems' capabilities have been described in general terms. This first part has been aimed at the librarian whose primary concern is, "what will the automated system do for my library?" Part II, which follows, is a technical description of the programs, file structures, etc. of the two systems. It is intended for the data processor whose interest is technical and whose concern is the implementation of either system.

PART II

TECHNICAL DESCRIPTIONS

FILE ORGANIZATION

This part of the manual is devoted to the technical descriptions of the terminal system. The programs in the batch system can easily be written from the description in Part I and the print-outs which are illustrated in the appendices. Therefore, the technical detail of the batch system is not given. However, the terminal system is more complex; thus, it is necessary to describe the entire system.

Since random, or direct, access files are the basis for the terminal system, the file structure, construction, and manipulation are first explained. A description of the program structure follows. The source listings of the actual programs are appended to the section, and references to the symbolic program names are given as the programs are introduced.

File Structure[1]

The file structure of the terminal system was initially designed to attain a balance between file size and processing time. However, this balance could not be achieved because the amount of data which needed to be stored for each item was too great. Using a test set of fifty books, researchers found that approximately 2400 characters were needed to fully define each item. To keep processing time to a minimum, the file structure needed to be designed so that all information on a given item would be stored in (1) an index-sequential file which had an author key; (2) an index-sequential file with a title key; and (3) an index-sequential file with a subject key. Thus, an item could be rapidly searched by either author, title, or subject. Assuming that an item had only one entry in each file, with this file arrangement approximately 7200 characters would be needed to store each item. If this length record were used, the files would quickly overflow the available peripheral storage. Thus, researchers decided to concentrate on a minimum length record and sacrifice processing speed.

---

[1] Robert T. Divett, W. Wayne Jones, and Edwin S. Dahl; "A File Structure for an Automated Library Catalog." A paper presented to the Third International Congress on Medical Librarianship; May 5-9, 1969, Amsterdam, 9 pp. Much of the discourse on the files of the terminal system is based on this paper, but the paper has been extensivily revised.

A variable length record was needed to solve the file record problem. However, in August 1967, when this research was being done, IBM did not support a variable length record which used index-sequential file organization. Thus, a mixture of index-sequential and relative files was used to achieve the variable length record organization.

An asymmetric modular file structure which used the above file organization was selected for use in the terminal system. Conceptually, one module was established for each avenue of entry--author, title, and subject. Each module was constructed as an index of the entry points, a file of data elements, and a chain of loci of related elements. At the focus of these three modules was a common intermodular cross-reference list. The manipulation of these modules will be explained below.

File Construction

The direct access files for the terminal system were constructed from the IBM punched cards of all processing information which the library had maintained since its inception in 1963. All the necessary information for the files was contained in three sets of card data--the catalog cards, the shelf list cards, and the purchase order cards. (Fig. 12, 13, 14; pp. 83)

Programs which used these cards to build the files were first written. The cards data was transferred from the card sets to three separate tapes which were then merged into a single tape. This "merge tape," as the single tape is called, was sorted by accession number and processed to build a description file and an author file on random access storage. At the same time, a title tape and a subject tape were created. These tapes were then processed to build direct-access title and subject files.

The use and manipulation of these files and of the modules follows.

File Manipulation

In this section the individual members of the file set will be explained.

Accession File

The cross-reference list, known as the accession file, cannot be directly inspected by the inquiror. It is accessed through the module which is selected as the avenue of entry and is used to retrieve data relating to the same item but which is found in other modules. The accession file is relatively organized; and each record carries the accession number as its identification. This number, assigned during the acquisitions process, is unique to each item. It serves as a pointer to a description file which contains data or pointers to data not stored in the main file set. Included in this description file are pointers to the location file, the publisher file, and the vendor file. The accession file further contains pointers to the title and subject modules and serves as the chaining member of the author module. One additional function of the accession file is to carry, in its first record, counters showing the next available record position in each of the relative files in the file set.

The Author Module

The author module contains the author index and the author file. The former is an alphabetically-arranged list of surnames of authors which is index-sequentially organized. Each author named is accompanied by a pointer to the first occurence of that surname in the author file. The relatively-organized author file contains full names of authors, each with a pointer to the next occurence of the surname in the author file. Each name also carries a pointer to the first record in the accession file attributed to this author. That portion of the accession file which is the chaining member of the author module consists of four counters. The first counter is a pointer to the first author in the author file to whom the item is attributed. The second counter points to the next author credited with the work. This counter may be regarded as being in a rotary chain which is entered at any point and is completed when the pointer value is equal to the first value returned. The third counter merely distinguishes between main and additional authors relative to the work. The fourth counter points to the next accession record attributed to the author. This counter may be regarded as being in a linear chain which must be entered at the beginning and which is ended when a zero value is returned.

The Title Module

The title module is the simplest of the three modules, since only those titles which are perfectly known can be found. Searching techniques for approximately-presented or vaguely-known titles constitute a discipline which is beyond the scope of this project.

The title index is organized index-sequentially by title and includes pointers to the title file. The title file is a relative file containing data and pointers to the title chain. An interesting aspect of the title file is that notes or abstracts may be carried as unindexed titles for presentation to browsers. The chaining feature of the file set enables linking elements of indefinite length, even though the physical records are a fixed length.

The chaining member for the title module consists of several pointers--a pointer to the accession record to which the chain element relates, a pointer in linear chain to the accession record of the next item which is known by the same title, and a reflex pointer in rotary chain to the next title by which the given work is known. The title chain is ended when the first title is again indicated.

The Subject Module

The subject module is logically the same as the title module except that provision is made for a hierarchical structure of subject headings. This feature is based on the Medical Subject Headings (MeSH) of the National Library of Medicine. (See Fig. 11, p.82) It consists of a dictionary of two index-sequential files. One is orgainized alphabetically and contains the reference number by which each subject is also known. The other is organized according to the reference numbers and also contains pointers to a list of hierarchical declensions. The more general headings of a subject are identified by rounding the consecutive subfields of the reference number to zero from right to left and referring to the numeric dictionary with the resulting number. More specific topics are found by using the declension pointer to enter the declension list, then picking the numbers from the narrower catagories until a zero switch is encountered. A zero switch indicates the end of related headings at this logical level. Searching by and for subject is like that described for title searching and bears no further discourse.

In General

Careful analysis of the file concepts presented here show this method of indexing to be flexible, simple, and replete with potential. In fact, the greater the number of items cataloged and in the file, the more efficient the files become. As an example, thousands of items cataloged with the same subject may be on file, yet the common subject appears only once. It is also apparent that through the chain concept, a virtually infinite number of tracings may be indexed to a given work while the system retains the intrinsic machine economy of the fixed length record. In short, the file structure seems eminently suited for the small to medium library for which it was developed.

Searching

As an example of how these files are used in the context of the library, a demonstration of their use in searching in a prepared data base is given below. In library searching, the logical avenues of entry are those of author, title, and subject. Ideally, in the automated system, these could be presented in any "and," "or," and "not" logical combination. In the well-designed, automated inquiry system, the recall and spurious recall, or fallout, values would also be manipulatable by the querist through use of "and/or" gates, acceptance thresholds, or other logical devices. Thus, these logical mechanisms have been provided in this system.

In a typical author search the name of the author is presented, last name first followed by the first and middle names or initials. The name is stripped to surname and passed through the author index using the surname as the key. The result of this search is a record containing the given surname and a pointer to the author file. The pointer to the author file is now used to fetch the appropriate record in the relative author file. The resulting record contains an author's full name, a pointer to another record in this file relating to another person with the same surname, and a pointer to the cross-reference, or accession file. At this time, the full name just acquired is compared to the given author's full name. If an unequal comparison results, the pointer to another name having the same surname is used to fetch another record; and the name test is repeated. This sequence is repeated until a positive comparison is made or the pointer to the next author returns to zero, at which it becomes apparent that the requested author is not in the file.

When an equal name comparison is detected, the pointer to the accession file is used to fetch the appropriate accession record. Through pointers to other files, the accession record will then obtain title(s), subject(s), description, publisher, vendor, and location of the item by the given author. The last pointer in the accession record points to an additional record in the accession file pertaining to another book written by the given author. This pointer is used to chain down the accession file obtaining all citations for the given author. This can be repeated until the pointer to the next book becomes zero.

Each time an accession record is fetched, nine pointers to access other files are obtained. These pointers are: (1) accession number [containing (2) a pointer to the publisher file and (3) a pointer to the location file]; (4) pointer to the title file; (5) pointer to the subject file; (6) pointer to the author file [not used when entering via an author search]; (7) pointer to a joint author; (8) pointer indicating author level as either the main author or a joint author; and (9) pointer to the next book by this author.

Using pointer (4) to access a record in the title file obtains a record containing a title and a pointer to the title chain. The title just obtained is a title of the book by a given author. It must now be determined if another title for the item exists. Using the pointer to the title chain to acquire a title chain record emits a pointer to another title for this book. If this pointer is the same as the title file record number previously obtained, no other title exists for this book. Otherwise the pointer to the other title is used to access another record in the title file, and the above procedure is repeated until all titles have been detected. Pointer (5) is used in conjunction with the subject file and the subject chain, obtained in the subject file, will ultimately bring forth all subjects associated with the given book.

PROGRAM STRUCTURE

Program Languages

The programs for the terminal system were written in a combination of programming languages. The specifications of the grant required that the system be developed in a higher level programming language for ease of conversion from one machine manufacturer to another. Since COBOL seems to have the greatest degree of compatibility for all machine manufacturers, it was used as much as possible. However, problems were encountered which could not be solved by using COBOL, thus FORTRAN and assembly languages were used in addition. FORTRAN was used on all the input-output routines which were required to manipulate the relatively organized files. Assembly was used to write the following routines:

(1) the input-output interface which required extensive use of the BTAM macros,
(2) the terminal monitors which required use of the attach macro and extensive bit maniputation, and
(3) IØENQ routines which used ENQ and DEQ macros.

The system also required tables which were more easily estabished by using assembly language.

Since the user of this system may wish to convert the FORTRAN or assembly programs to COBOL or other languages, the programs written in these languages are listed below by subroutine name.

Assembly

- FLNMLST
- IØENQ
  - CBLK SECT
  - IØ2260
  - TRNSLT TABLES
- IØTABL
- TERM MACRO

FØRTRAN

ACNFØR
AUTFØR
AUTSEQ
DNTREE
DSCSEQ
RAUTIØ
RDALP
RDNUM
RDPLACE
RDPLC
RDPUB
RDPUBL
RDSCIØ
RDSCRT
RDVEN
READFILEF
REWDSC
REWSUB
RSUBIØ
RTITIØ
RUAUT
RWACNU
RWACN1
RWAUTU
RWSB1U
RWSB2U
RWTIT
RWT1UP
RWT2UP
SUBSEQ
SU1FØR
SU2FØR
TITSEQ
TI1FØR
TI2FØR
UPTREE
WAUTIØ
WDSCIØ
WHATFØRMSUBR
WTACN
WTAUT
WTITIØ
WTSCRI
WTSUBIØ
WTSUB1
WTSUB2
WTTI1
WTTI2

Resident Programs

The following are resident programs and remain in core storage at all times. Each is discussed separately.

(1) Input-Output Interface for the IBM 2260's (IØ2260)
(2) Terminal Monitor
(3) Direct Access Input-Output and Open-Close Routines
(4) Table
(5) IØENQ Subroutines (IØENQ)

Input-Output Interface for the IBM 2260's

This program was written specifically for the equipment configuration at this particular library. The equipment included a remote IBM 2848 model 2, a leased line, and two IBM 2260's. If any other equipment configuration were used, some program modification would need to be made.

The function of this program is, as its name indicates, a program which controls all the input to and the output from the 2260's, but it also has other functions. It polls each 2260 every three seconds to determine if there is an active terminal; it passes all information accepted as either input or output through a translate table; it detects any line or transmission error; and it detects which terminal is responding and gives control to the appropriate terminal monitor. I/02260, as this program is called, is in core memory at all times.

Terminal Monitor

For each 2260 in the system there must be a terminal monitor residing in core storage. If sixteen terminals are attached, then sixteen terminal monitors must be in storage at all times. Due to some of the lengthy tables within each terminal monitor, a considerable portion of the allotted core storage would be used for the terminal monitors if a large number of 2260's were attached to the system. Thus, it is imperative that the network of programs be modified to involve only one terminal monitor; but because of the time limitation of the grant and because storage space for only two 2260's did not present a problem, no further work was done in this area.

The function of the terminal monitor program is to examine the input from the 2260 for the control words "order," "receive," "catalog," or "inquire." (The input from the 2260 has been read into an input-output buffer by the I/Ø2260 program.) If the terminal monitor detects a control word, the processing module number which corresponds to the word is placed in the calling sequence of an attach macro. The attach macro is then executed and control is passed to the module which was loaded when the attach macro was executed.

If one of the control words is not detected in the examination of the input from the 2260, the terminal monitor then examines a word in the event control block to determine the module number which is to be executed next. The appropriate parameter of the attach macro parameter list is loaded with the module number and control is then passed to that module.

One of the main functions of the processing modules is to set the word in the event control block to indicate the module number which should be executed next. (This will be further explained in the Transient Program section.) The processing module also sets switches in the event control block word to indicate the next I/Ø function of the I/Ø2260 program. The possible I/Ø functions are read next full buffer, read next manual input, write next, or process.

The processing module then returns control to the terminal monitor at the instruction following the execution of the attach macro. At this point, the terminal monitor returns control to the I/Ø2260 program. I/Ø2260 now has the necessary information to determine its next operation.

The preceding sequence of events is then repeated until the task is completed.

Direct Access Input-Output and Open-Close Routines

Since there are sixteen files associated with this system, the time required to open or close then is significant, especially since all the index-sequential files are opened as input and as I/0 files simultaneously. Thus, to reduce the open-close time, the open routine is the first executed each day when the system is put into operation. The files then remain open until they are closed when the system is terminated for the day. The close routine is the last routine executed.

The direct access I/0 routines are each assigned a routine number. (See following table.) None of these input-output routines can be directly entered from one of the processing modules. Instead the appropriate routine number is placed in the calling sequence of the I0ENQ subroutine, and the input-output routine is entered. (See subsection on the I0ENQ subroutine.)

Direct access input-output buffer areas are defined within each terminal monitor. Since the terminal monitors are resident in core storage at all times, the information in the direct access buffer is also in core storage and can be passed from one processing module to another.

Table

The following is a table of the I/0 routine numbers and names. The numbers are assigned by the relative position of the name in the table, and the names are the symbolic names which have been assigned.

Number	Symbolic Name	Function of the Program
0	AUTSEQ	Reads author index-sequential file
1	TITSEQ	Reads title index-sequential file
2	SUBSEQ	Reads subject index-sequential file
3	DSCSEQ	Reads descriptive index-sequential file
4	ACNF0R	Reads accession F0RTRAN file
5	AUTF0R	Reads author F0RTRAN file
6	TI1F0R	Reads title 1 F0RTRAN file
7	TI2F0R	Reads title 2 F0RTRAN file
8	SU1F0R	Reads subject 1 F0RTRAN file
9	SU2F0R	Reads subject 2 F0RTRAN file
10	RDVEN	Reads vendor F0RTRAN file
11	RDPLC	Reads place F0RTRAN file
12	RDPUB	Reads publisher F0RTRAN file
13	RDALP	Reads alpha dictionary index-sequential file
14	RDNUM	Reads numeric dictionary index-sequential file
15	RFILEF	Reads dictionary F0RTRAN file
16	RDSCRT	Reads scratch file
17	WTSCRT	Writes scratch file
18	NOT TO BE USED	Pointer to FLNMLST
19	NOT TO BE USED	Pointer to scratch file control block
20	RDPLACE	Reads place file given place (alpha)
21	RDPUBL	Reads publisher file given publisher (alpha)
22	(multiple entry)	
	(1) RAUTI0	Reads author (ISAM) opened I/0
	(2) REWAUT	Rewrites author (ISAM) opened I/0
	(3) WAUTI0	Writes author (ISAM) opened I/0
23	(1) RTITI0	Reads title (ISAM) opened I/0
	(2) REWTIT	Rewrites title (ISAM) opened I/0
	(3) WTITI0	Writes title (ISAM) opened I/0
24	(1) RSUBI0	Reads subject (ISAM) opened I/0
	(2) REWSUB	Rewrites subject (ISAM) opened I/0
	(3) WSUBI0	Writes subject (ISAM) opened I/0
25	(1) RDSCI0	Reads descriptive (ISAM) opened I/0
	(2) REWDSC	Rewrites descriptive (ISAM) opened I/0
	(3) WDSCI0	Writes descriptive (ISAM) opened I/0
26	RWACN1	Reads and writes first record in accession file
27	WTAUT	Writes author (F0RTRAN)
28	WTSUB1	Writes subject 1 (F0RTRAN)
29	WTSUB2	Writes subject 2 (F0RTRAN)
30	WTTI1	Writes title 1 (F0RTRAN)
31	WTTI2	Wrires title 2 (F0RTRAN)
32	WTACN	Writes accession (F0RTRAN) except first record
33	RWACNU	Reads and writes accession (F0RTRAN) in update mode
34	RWAUTU	Reads and writes author (F0RTRAN) in update mode
35	RWT1UP	Reads and writes title 1 (F0RTRAN) in update mode
36	RWT2UP	Reads and writes title 2 (F0RTRAN) in update mode
37	RWSB1U	Reads and writes subject 1 (F0RTRAN) in update mode
38	RWSB2U	Reads and writes subject 2 (F0RTRAN) in update mode

I0ENQ Subroutine

Since many people may be using this system simultaneously, it is imperative that certain input-output routines, primarily any routines which are involved in updating the files, are not executed at the same time. The function of the I0ENQ subroutine is to eliminate this possibility.

As stated before, when a processing module is required to transfer output to a disk or to obtain input from a disk, it passes certain parameters to the I0ENQ subroutine. These parameters are: (1) the required relative routine number, (2) the address of the subroutine table, (3) the address of the direct access buffer located in the terminal monitor, and (4) the record key required for this input-output operation.

Using the first parameter in the I0ENQ calling sequence, the relative routine number, as a resource name, the I0ENQ can construct a queue on the resource. As a call for the I/0 routine (resource) is encountered, the resource queue is examined for activity. If it is active, the request is stacked at the end of the request line. When the active request is completed, the resource is released; and the elements in the queue are shifted one position. No priority constraints are involved in this procedure; it is strictly on a first-in, first-out basis.

After a requested I/0 operation gains control of the resource, the address of the buffer and the record key are passed to the appropriate input-output subroutine; and the input-output function is completed. Control is then returned to the I0ENQ subroutine which releases the resource and returns control to the processing module.

This programming method was written for use with both multiprogramming for a fixed number of tasks (MFT) and for a variable number of tasks (MVT). However, since a computer operating in an MVT environment was not available, success of the method in an MVT environment has not been checked.

Transient Programs

The transient programs for the terminal system consist of forty-eight modules which were written in small segments to gain a degree of multiprocessing for the system and to insure that any program in core storage at any time would not overflow the core boundaries allotted for the system.

Ideally, a terminal system for library processing would use IBM's "roll-in, roll-out" feature in which a program could be "rolled-in" to core storage, partially executed, and "rolled-out" onto a storage device. Another program, or the same program, could then be "rolled-in" and executed from the beginning if it is a new program, or from the point where it was "rolled-out" if it is a program which was only partially executed. This feature would have been used for this system, but IBM did not announce it until after the project had begun. It is still not completely operational. Therefore, the programs for the system were written in small segments so the modules could be called into core storage, executed, and removed from core memory in one quick in-out operation. The segments had to be small so the programs would not overflow the storage allotted for the terminal.

Once a program was removed from storage, another terminal could gain control of the system and perform another task. Thus the system attained a degree of multiprocessing.

The modules within the system vary in length. Though core storage limits the maximum length of each module, there are no minimum length requirements. The length of each module is determined by the function which it performs. A module accepts a statement from the 2260 and processes it until something is output to the 2260. The processing stops and the module ends when an output function is performed.

During the time the module is waiting for the 2260 I/0, another user is brought in and uses storage.

Each library function requires the use of several modules. The table below illustrates which modules are used for each function.

Function	Module numbers
Order	50,51,52,53,54,55,56,57,99.
Receive	60,61,99.
Catalog	70,71,72,73,74,75,76,77,78,79,80,81, 82,83,84,85,86,87,88,89,91,92,93,94, 95,99.
Author Search	1,2,3,10,11,99.
Title Search	1,2,4,10,11,99.
Subject Search	1,2,5,10,11,16,17,18,19,20,99.

From the above table it is seen that module 99 is common to all processing functions. Module 99 terminates each function and brings the system back to the beginning.

Modules 1, 2, and 10 are common to the search functions. Module 1 controls the options by which a user can search, module 2 is the switching module for the options, and module 10 controls all the screen displays. Module 10 also has a limitation built into it. Only 13,225 citations can be displayed in one sequence, but it was felt that this number was sufficient to cover the number any one terminal user would want to view at one time.

Though the system was initially designed to accomodate 100 modules, only forty-eight were used. The remaining modules would be used for circulation and for special uses of the system.

As an example of how these modules work, a simple author search is explained below. Details on the other functions can be obtained from the program listings in the appendices.

Until the terminals of the system are activated by a library user, the system is in a static mode with the terminal monitor program and the I/02260 program in control. No processing modules are in core storage at this time though the terminal monitor and I/02260 programs are. The system is static with "BEGIN INQUIRY" (2260-65) on the terminal and with the switches in the event control block set to load mod 1 and to read manual input as the next I/0 operation.

When a user types in his borrower's indentification number and depresses the enter-shift keys, I/02260 passes control to the appropriate terminal monitor which loads mod 1 and passes control to it. Mod 1 constructs the screen display:

```
YOU MAY SEARCH BY AUTHOR, TITLE, SUBJECT
OR ANY COMBINATION THEREOF.  ENTER YOUR
CATAGORY FROM ONE OF THE FOLLOWING
A,T,S,ATS,AS,TS, OR AT.          (2260-66)
```

Mod 1 also sets the event control block switches to write full buffer,

to read manual input next, and to load mod 2. Mod 1 then returns control to the terminal monitor which returns control to I/02260 which displays the message and begins polling for the next activity.

For an author search the user types in "A" and depresses the enter-shift keys. I/02260 accepts the reply, passes control to the terminal monitor which loads mod 2 and passes control to it.

Mod 2 is the switching module which analyses the message from the terminal, determines what type of search it is, and constructs the appropriate message. In the author search it constructs:

```
ENTER YOUR REQUESTED AUTHOR.  NOT MORE
THAN 213 CHARACTERS PLEASE.  (2260-67)
```

It also sets the event congrol block switches to write full buffer, read manual input next, and load mod 3. (Switches to load other modules would be set if the search was another type.) Mod 2 then returns control to the terminal monitor which returns control to I/02260 which displays the message and polls the terminal for activity.

The user then types in the requested author and depresses the enter-shift keys. I/02260 accepts the input and passes control to the terminal monitor which loads mod 3 and passes control to it.

Mod 3 accepts the author message and interrogates the index-sequential author file. If it detects no matching author record in the file, mod 3 constructs the message:

```
THERE IS NO INFORMATION FOR YOUR REQUEST.
TRY AGAIN.                          (2260-68)
```

It then sets the event control block switches to write full buffer, read manual input next, and load mod 3. Control is passed to I/02260 through the terminal monitor, and the process is repeated with another user response.

If a matching author is detected, mod 3 chains through the accession file obtaining all citations relative to the request and inserting these accession number pointers into a table in the terminal monitor communication area. If more than three pointer are obtained, mod 3 constructs the message:

```
THERE ARE X CITATIONS RELATIVE TO YOUR
REQUEST.  WOULD YOU CARE TO FURTHER DEFINE
YOUR REQUEST WITH THE FOLLOWING OPTIONS:
BOOKS ONLY...JOURNALS ONLY...
LANGUAGE...DATE OF PUBLICATION YY TO YY.
                                (2260-46)
```

It then sets the event control block switches to write full buffer, read manual input next, and load mod 11. Mod 3 then returns control to the terminal monitor which passes control to I/02260 which in turn displays the message and polls for activity.

If the user responds with one of the options, mod 11 interrogates all citations deleting those pointers not relative to the redefined request. Control is then passed to mod 10.

If the user does not wish to redefine his search, control is immediately passed to mod 10.

If three citations or less are obtained when mod 3 chained through the accession file, mod 3 sets the event control block switches to write full buffer, process, and load mod 10. The term "process" means that control is immediately passed to mod 10. No user response is required. Mod 3 then returns control to the terminal monitor which loads mod 10 and passes control to it.

Using the table of pointers constructed by mod 3, mod 10 obtains a citation for each pointer, constructs a citation, sets the event constrol block switches to write full buffer, process, and load mod 10. Control is returned to I/02260 through the terminal monitor and the citation is displayed. The user depresses the enter-shift keys, and the process is completed until the table is exhausted. Mod 10 then constructs the message:

```
THERE ARE NO MORE CITATIONS RELATIVE
TO YOUR REQUEST.          (2260-48)
```

Mod 10 then sets the event control block switches to write full buffer, read manual input next, and load mod 99.

Once mod 99 has been executed, the system returns to the static mode with "BEGIN INQUIRY" on the 2260.

If anywhere in the search process or in any other function of the system the user wants to stop the system, he can type "/STOP" as a response and enter it. The system will accept the message and be brought back to the beginning.

Summary

In the foregoing description the basic file organization and program structure of the terminal system have been discussed. As an example of the files and programs, a simple author search has been used to illustrate the operation of the system. Further technical detail on any point can be obtained from the source listings of the programs in the appendices.

BIBLIOGRAPHY

Ausherman, Marian R., William D. Wittekind, and Robert T. Divett, "PIL - A Control Procedure for Technical Services Management," *Bulletin of the Medical Library Association*, 55:394-398, *October, 1967*.

Divett, Robert T., "Mechanization in a New Medical School Library - I. Acquisitions and Cataloging," *Bulletin of the Medical Library Association*, 53:15-25, January, *1965*.

Divett, Robert T., W. Wayne Jones, and Edwin S. Dahl, "A File Structure for an Automated Library Catalog," A paper presented to the Third International Congress on Medical Librarianship; May 5-9, 1969, Amsterdam, 9pp.

Grosch, Audrey N., "Implication of On-Line Systems Techniques for a Decentralized Research System," *College and Research Libraries*, 30:112-118, March, 1969.

"Oregon's Computer Instruction Networks," *Educom, Bulletin of the Interuniversity Communications Council*, 3:1-2, 1968.

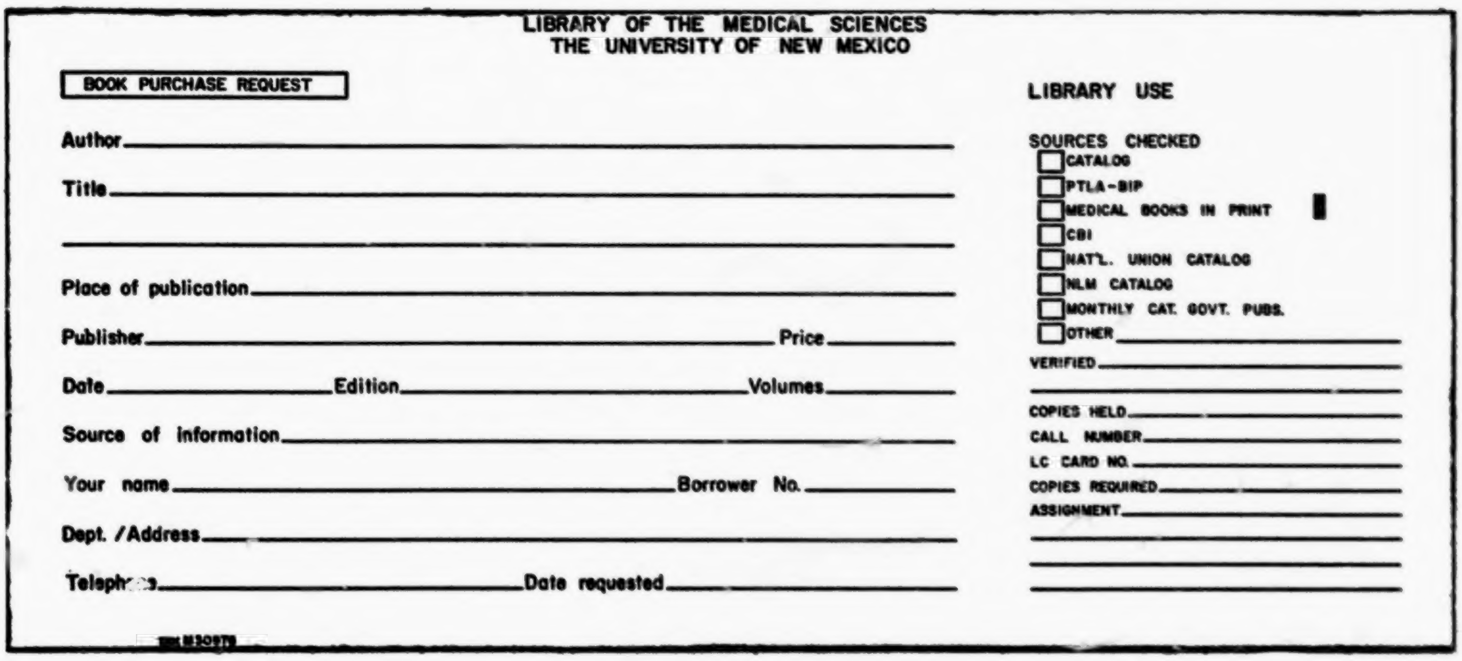
Orr, Richard H. and Alice A. Leeds, "Biomedical Literature: Volume, Growth, and Other Characteristics," *Federation Proceedings*, 23:1310-1331, 1964.

Payne, Ladye M., Louise Small, and Robert T. Divett, "Mechanization in a New Medical School Library - II. Serials and Circulation," *Bulletin of the Medical Library Association*, 54:337-350, *October, 1966*.

APPENDICES

APPENDIX A

LIBRARY FORMS



LIBRARY OF THE MEDICAL SCIENCES
THE UNIVERSITY OF NEW MEXICO

BOOK PURCHASE REQUEST

Author__________

Title__________

____

Place of publication__________

Publisher__________ Price__________

Date__________ Edition__________ Volumes__________

Source of information__________

Your name__________ Borrower No.__________

Dept. /Address__________

Telephone__________ Date requested__________

[illegible]

LIBRARY USE

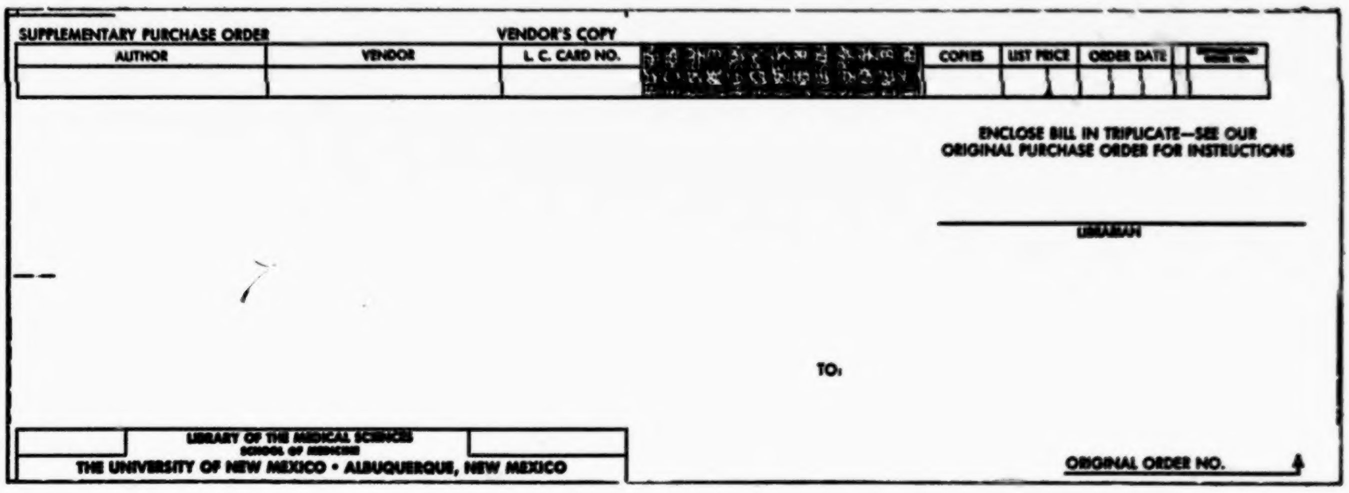
SOURCES CHECKED
☐ CATALOG
☐ PTLA-BIP
☐ MEDICAL BOOKS IN PRINT
☐ CBI
☐ NAT'L. UNION CATALOG
☐ NLM CATALOG
☐ MONTHLY CAT. GOVT. PUBS.
☐ OTHER__________

VERIFIED__________
____

COPIES HELD__________
CALL NUMBER__________
LC CARD NO.__________
COPIES REQUIRED__________
ASSIGNMENT__________
____
____
____

Figure 1

Purchase Request Form



SUPPLEMENTARY PURCHASE ORDER

VENDOR'S COPY

AUTHOR	VENDOR	L. C. CARD NO.	[redacted]	COPIES	LIST PRICE	ORDER DATE		[illegible]

ENCLOSE BILL IN TRIPLICATE—SEE OUR ORIGINAL PURCHASE ORDER FOR INSTRUCTIONS

____
LIBRARIAN

TO:

LIBRARY OF THE MEDICAL SCIENCES
SCHOOL OF MEDICINE
THE UNIVERSITY OF NEW MEXICO • ALBUQUERQUE, NEW MEXICO

ORIGINAL ORDER NO.__________

Figure 2

Supplemental Order Form

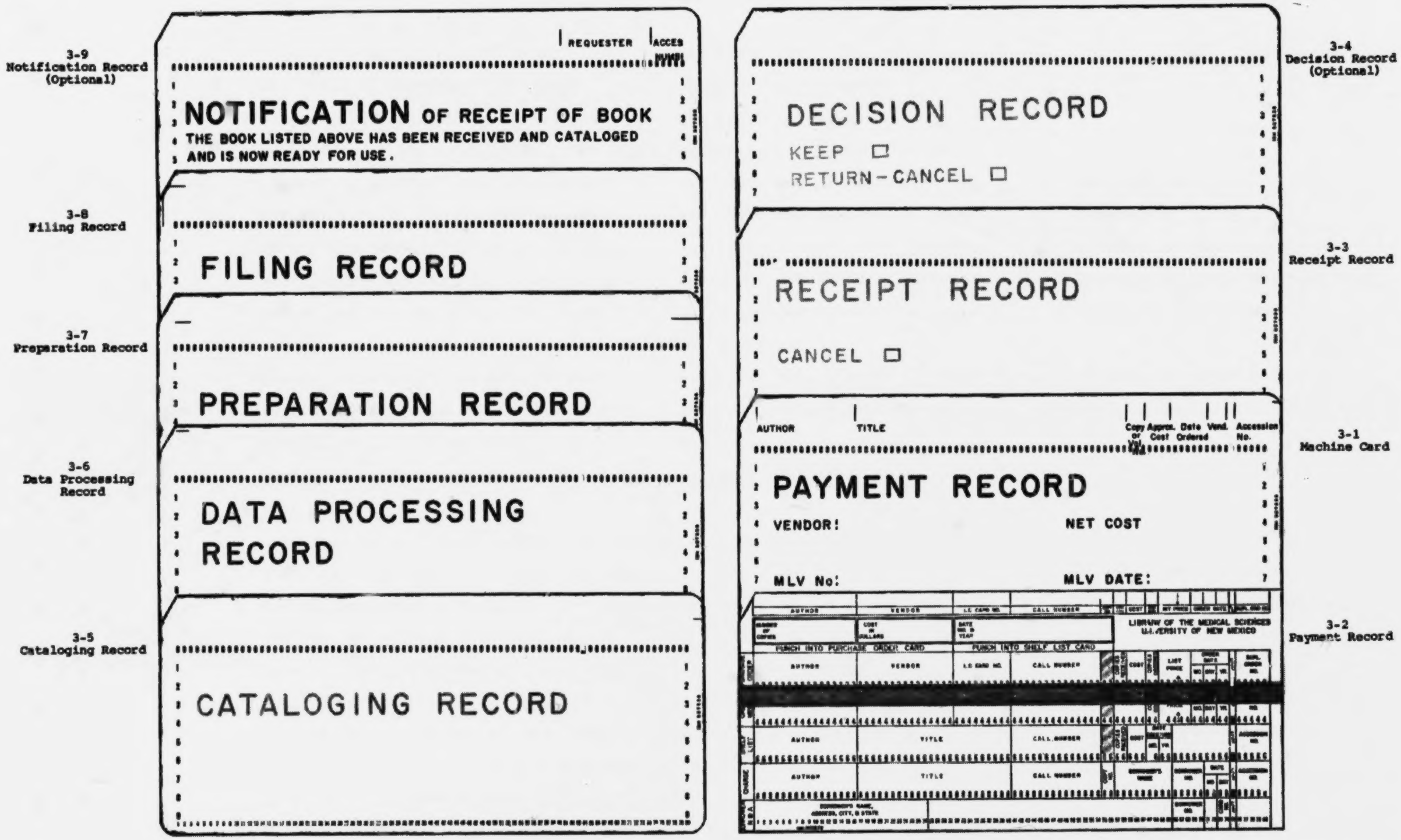
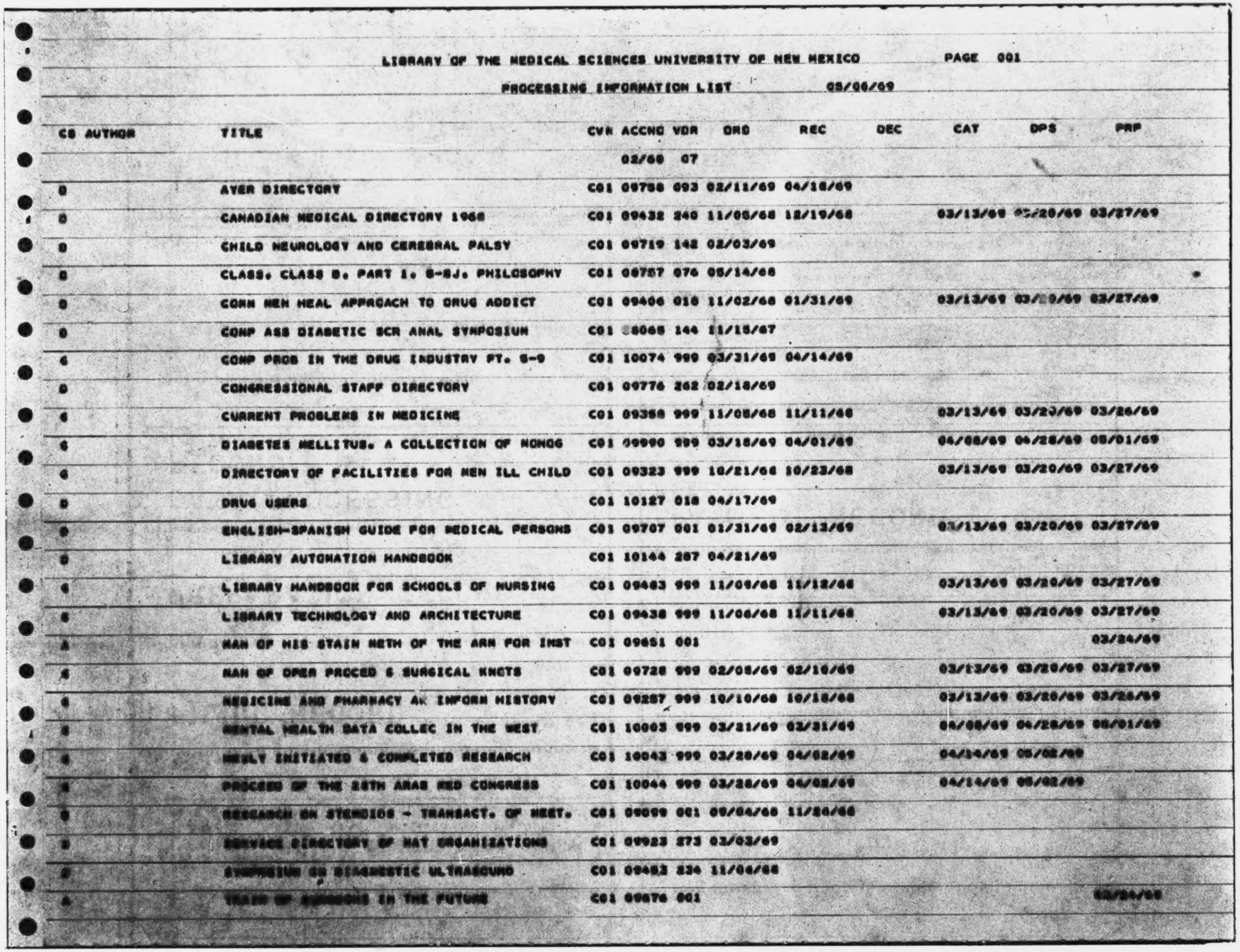

3-9 Notification Record (Optional)
3-8 Filing Record
3-7 Preparation Record
3-6 Data Processing Record
3-5 Cataloging Record
3-4 Decision Record (Optional)
3-3 Receipt Record
3-1 Machine Card
3-2 Payment Record
REQUESTER
ACCES
NOTIFICATION OF RECEIPT OF BOOK
THE BOOK LISTED ABOVE HAS BEEN RECEIVED AND CATALOGED AND IS NOW READY FOR USE.
FILING RECORD
PREPARATION RECORD
DATA PROCESSING RECORD
CATALOGING RECORD
DECISION RECORD
KEEP ☐
RETURN-CANCEL ☐
RECEIPT RECORD
CANCEL ☐
AUTHOR
TITLE
Copy or Vol.
Approx. Cost
Date Ordered
Vend.
Accession No.
PAYMENT RECORD
VENDOR:
NET COST
MLV No:
MLV DATE:
AUTHOR
VENDOR
LC CARD NO.
CALL NUMBER
COST IN DOLLARS
PUNCH INTO PURCHASE ORDER CARD
PUNCH INTO SHELF LIST CARD
LIBRARY OF THE MEDICAL SCIENCES UNIVERSITY OF NEW MEXICO
TITLE

Figure 3

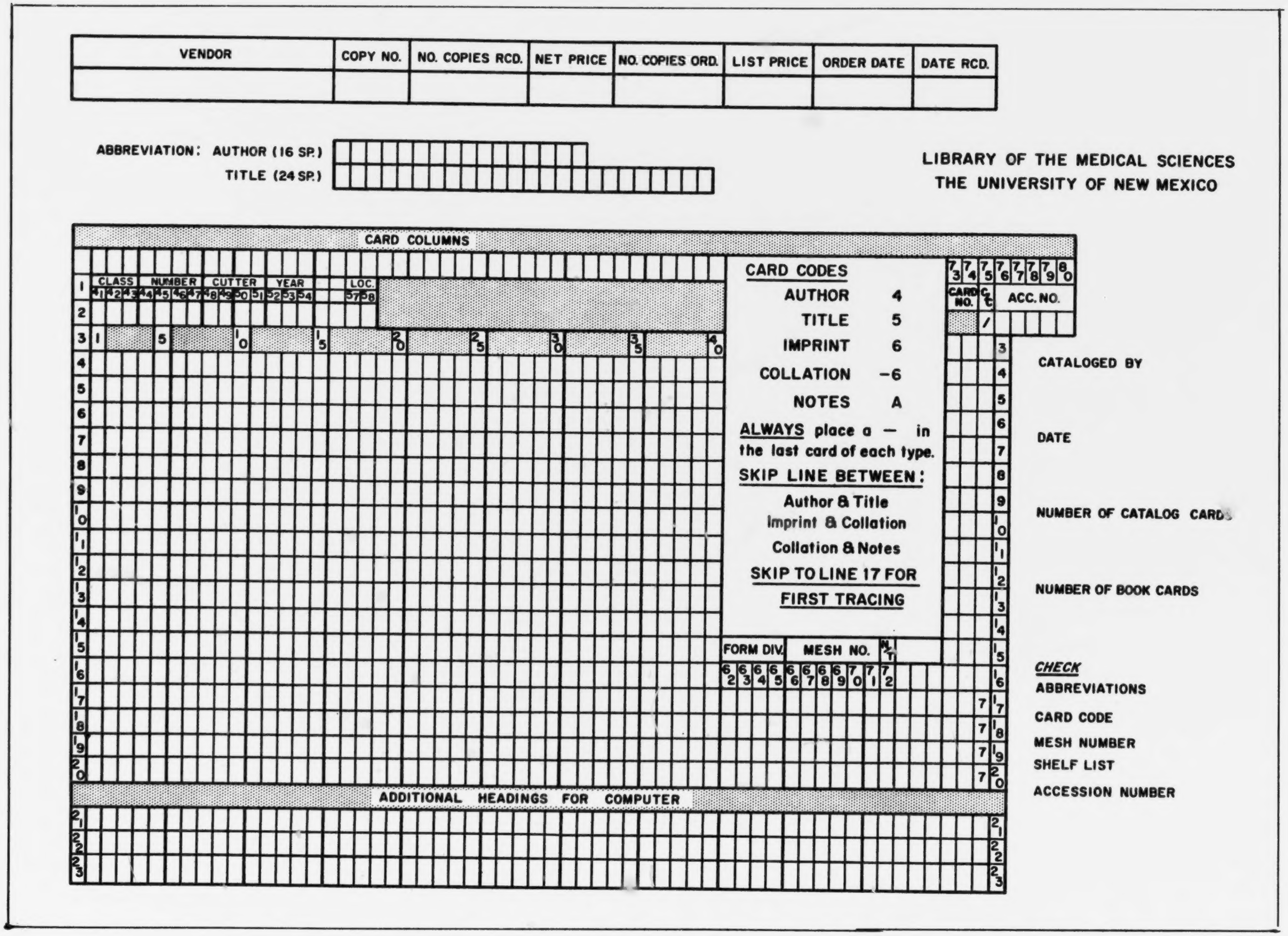
PIL Decklet



```
                              LIBRARY OF THE MEDICAL SCIENCES UNIVERSITY OF NEW MEXICO          PAGE  001

                                           PROCESSING INFORMATION LIST          05/06/69

CS AUTHOR         TITLE                                    CVN ACCNO VDR   ORD       REC       DEC       CAT       DPS       PRF

                                                               02/68  07

D                 AYER DIRECTORY                           C01 09758 093 02/11/69 04/18/69
D                 CANADIAN MEDICAL DIRECTORY 1968          C01 09432 240 11/05/68 12/19/68           03/13/69 03/20/69 03/27/69
D                 CHILD NEUROLOGY AND CEREBRAL PALSY       C01 09719 142 02/03/69
D                 CLASS. CLASS B. PART 1. B-BJ. PHILOSOPHY C01 08757 076 05/14/68
D                 COMM MEN HEAL APPROACH TO DRUG ADDICT    C01 09406 018 11/02/68 01/31/69           03/13/69 03/20/69 03/27/69
D                 COMP ASS DIABETIC SCR ANAL SYMPOSIUM     C01 08068 144 11/15/67
G                 COMP PROB IN THE DRUG INDUSTRY PT. 5-9   C01 10074 999 03/31/69 04/14/69
D                 CONGRESSIONAL STAFF DIRECTORY            C01 09776 262 02/18/69
G                 CURRENT PROBLEMS IN MEDICINE             C01 09358 999 11/05/68 11/11/68           03/13/69 03/20/69 03/26/69
G                 DIABETES MELLITUS. A COLLECTION OF MONOG C01 09990 999 03/18/69 04/01/69           04/08/69 04/28/69 05/01/69
G                 DIRECTORY OF FACILITIES FOR MEN ILL CHILD C01 09323 999 10/21/68 10/23/68          03/13/69 03/20/69 03/27/69
D                 DRUG USERS                               C01 10127 018 04/17/69
D                 ENGLISH-SPANISH GUIDE FOR MEDICAL PERSONS C01 09707 001 01/31/69 02/13/69          03/13/69 03/20/69 03/27/69
D                 LIBRARY AUTOMATION HANDBOOK              C01 10144 287 04/21/69
G                 LIBRARY HANDBOOK FOR SCHOOLS OF NURSING  C01 09483 999 11/09/68 11/12/68           03/13/69 03/20/69 03/27/69
G                 LIBRARY TECHNOLOGY AND ARCHITECTURE      C01 09438 999 11/06/68 11/11/68           03/13/69 03/20/69 03/27/69
A                 MAN OF HIS STAIN METH OF THE ARM FOR INST C01 09651 001                                                 03/24/69
A                 MAN OF OPER PROCED & SURGICAL KNOTS      C01 09728 999 02/05/69 02/10/69           03/13/69 03/20/69 03/27/69
G                 MEDICINE AND PHARMACY AN INFORM HISTORY  C01 09257 999 10/10/68 10/18/68           03/13/69 03/20/69 03/26/69
G                 MENTAL HEALTH DATA COLLEC IN THE WEST    C01 10003 999 03/21/69 03/31/69           04/08/69 04/28/69 05/01/69
G                 NEWLY INITIATED & COMPLETED RESEARCH     C01 10043 999 03/20/69 04/02/69           04/14/69 05/02/69
G                 PROCEED OF THE 25TH ARAB MED CONGRESS    C01 10044 999 03/28/69 04/02/69           04/14/69 05/02/69
D                 RESEARCH ON STEROIDS - TRANSACT. OF MEET. C01 09099 001 09/04/68 11/20/68
D                 SERVICE DIRECTORY OF NAT ORGANIZATIONS   C01 09923 273 03/03/69
D                 SYMPOSIUM ON DIAGNOSTIC ULTRASOUND       C01 09483 234 11/04/68
A                 TRAIN OF SURGEONS IN THE FUTURE          C01 09876 001                                                 03/24/69
```

Figure 4

Processing Information List



VENDOR	COPY NO.	NO. COPIES RCD.	NET PRICE	NO. COPIES ORD.	LIST PRICE	ORDER DATE	DATE RCD.

ABBREVIATION: AUTHOR (16 SP.)
TITLE (24 SP.)

LIBRARY OF THE MEDICAL SCIENCES
THE UNIVERSITY OF NEW MEXICO

CARD COLUMNS

CLASS NUMBER CUTTER YEAR LOC.

CARD NO. C ACC. NO.

CARD CODES

AUTHOR	4
TITLE	5
IMPRINT	6
COLLATION	-6
NOTES	A

ALWAYS place a — in the last card of each type.

SKIP LINE BETWEEN:
Author & Title
Imprint & Collation
Collation & Notes

SKIP TO LINE 17 FOR FIRST TRACING

FORM DIV. MESH NO.

CATALOGED BY

DATE

NUMBER OF CATALOG CARDS

NUMBER OF BOOK CARDS

CHECK
ABBREVIATIONS
CARD CODE
MESH NUMBER
SHELF LIST
ACCESSION NUMBER

ADDITIONAL HEADINGS FOR COMPUTER

Figure 5

Catalog Worksheet

```
                        MONOGRAPH SHELF LISTING

              ENCYCLOPEDIA AMER      AE 0027 EN01 948  30  08590

BUTTRESS, F.  WORLD LIST OF ABBREV   AS 0008 B 97 966      06981

              WORLD OF LEARNING      AS 0008 W 89 963      00129

SATRE, J. P.  BEING & NOTHINGNESS    B  0819 SA07 956      G0653
```

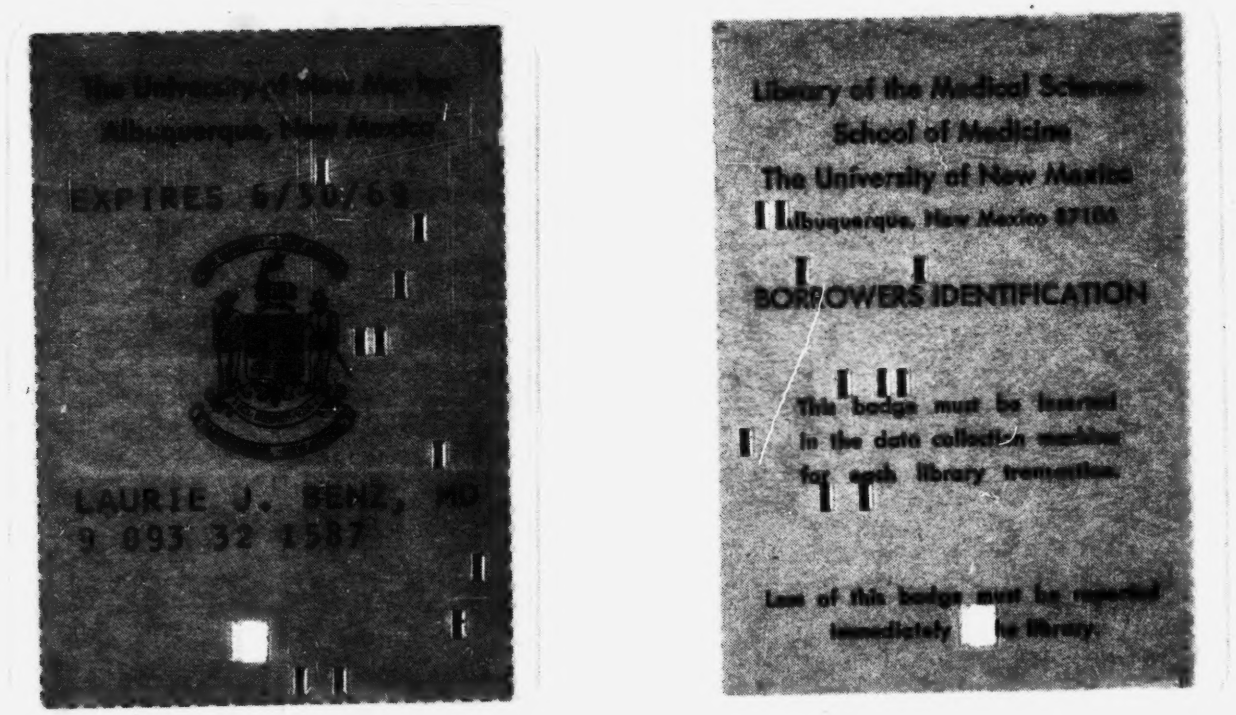
Figure 6

Monograph Shelf List

```
CURRENT SERIALS RECEIVED

ACTA BIOL MED GERMAN     10, 1963-15, 1966               1  00415
                         16=1-6, 1966 17=1-6, 1966       2  00415
                         18=1-6, 1967 19=1-6, 1967       3  00415
                         20=1-6, 1968 21=1-3, 1968       4  00415


CURRENT SERIALS RECEIVED

MARCH 17, 1969

TITLE                   VOLUME  ISSUE  PART  YEAR  ACCNO  DATE-IN   OWNER

ACTA BIOL MED GERMAN      21      4     *    ****  00415  03/17/69    S
ACTA BIOL MED GERMAN      21      5     *    ****  00415  03/17/69    S
ACTA BIOL MED GERMAN      21      6     *    ****  00415  03/17/69    S
```

Figure 7

Serials Shelf List - Cumulated and Supplementary



The University of New Mexico
Albuquerque, New Mexico
EXPIRES 6/30/69
LAURIE J. BENZ, MD
9 093 32 1587

Library of the Medical Sciences
School of Medicine
The University of New Mexico
Albuquerque, New Mexico 87106
BORROWERS IDENTIFICATION
This badge must be inserted in the data collection machine for each library transaction.
Loss of this badge must be reported immediately to the library.

Figure 8

Identification Badge

TYPE 0 - RESERVES

AUTHOR	TITLE	CALL NUMBER	AC-NO	BORROWER-N	TIME	DATE
ARNOLD, O.	SAVAGE SON	WZ 0100M 76951C01	03724	7777777777	2024	04/03/9
FED AMER SOC	EXMETABOLISM	QU 0120F 3198C01	09430	7000800088	2027	04/03/9

TYPE 3 - SERIALS

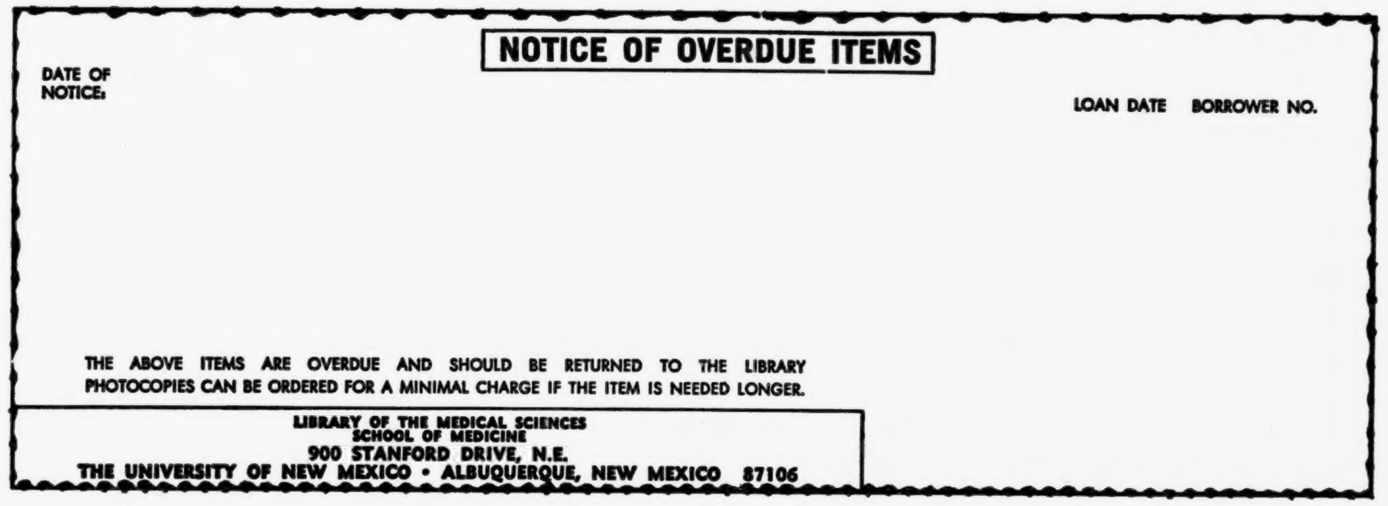
AUTHOR	TITLE	CALL NUMBER	AC-NO	BORROWER-N	DATE
ACTA MED SCAND		182 1967	09665	7000400044	05/06/9
ACTA NEUROL SCAND		45 1	08298	1295364081	05/05/9

TYPE 7 - MONOGRAPHS

AUTHOR	TITLE	CALL NUMBER	AC-NO	BORROWER-N	DATE
ARTZ, C.P.	TREATM OF BURNS	WO 0704AR07969C01	09665	3585306781	05/03/9
BAKER, A.A.	PSYCH DISORDERS	WM 0100B 17967C01	08298	9295364081	04/16/9

Figure 9

Circulation List



NOTICE OF OVERDUE ITEMS

DATE OF
NOTICE:

LOAN DATE BORROWER NO.

THE ABOVE ITEMS ARE OVERDUE AND SHOULD BE RETURNED TO THE LIBRARY
PHOTOCOPIES CAN BE ORDERED FOR A MINIMAL CHARGE IF THE ITEM IS NEEDED LONGER.

LIBRARY OF THE MEDICAL SCIENCES
SCHOOL OF MEDICINE
900 STANFORD DRIVE, N.E.
THE UNIVERSITY OF NEW MEXICO • ALBUQUERQUE, NEW MEXICO 87106

Figure 10

Overdue Form

PILONIDAL CYST (C2)

PIMELIC ACID (D2)

PINEAL BODY (A6, A8)
  X EPIPHYSIS CEREBRI (A6, A8)

PINEALOMA (C2)

PINK DISEASE see ACRODYNIA (C10)

PINNIPEDIA (B2)
  XU SEALS (B2)
  XU WALRUS (B2)

PINOCYTOSIS (G1)

PINTA (C1, C12)

PIPOCOLIC ACIDS (D2)

PITUITARY GLAND (A6)

A - ANATOMICAL TERMS

A6 - Endocrine System

ENDOCRINE GLANDS
  ADRENAL GLANDS
  ISLANDS OF LANGERHANS (A3)
  PARATHYROID GLANDS
  PINEAL BODY (A8)
  PITUITARY GLAND
  THYROID GLAND

ADRENAL GLANDS
  Adrenal Medulla
ADRENAL MEDULLA
CHROMAFFIN SYSTEM
  Adrenal Medulla
ENDOCRINE GLANDS
  Adrenal Glands
  Gonads (A5)
  Islands of Langerhans (A3)
  Parathyroid Glands
  Pineal Body (A8)

ENDOCRINE GLANDS
  (Continued)
  Pituitary Gland
  Thyroid Gland
GONADS (A5), etc.

Figure 11

National Library of Medicine Medical Subject Headings (MeSH)

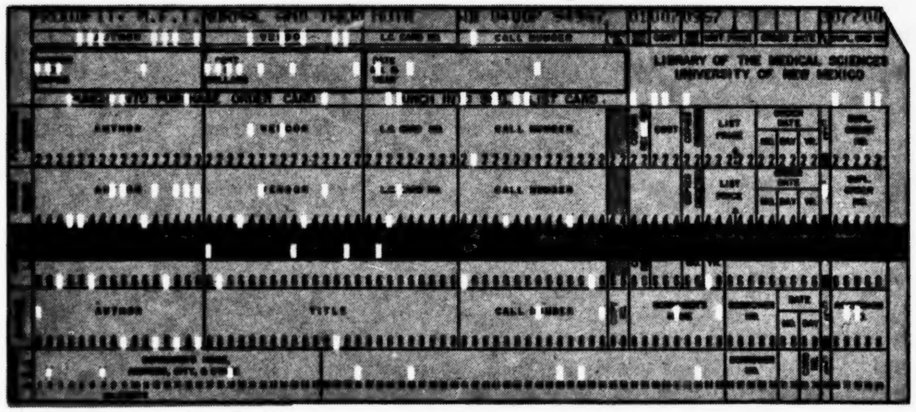

LIBRARY OF THE MEDICAL SCIENCES
UNIVERSITY OF NEW MEXICO
AUTHOR
VENDOR
LC CARD NO.
CALL NUMBER
TITLE

Figure 12

Shelf List Card

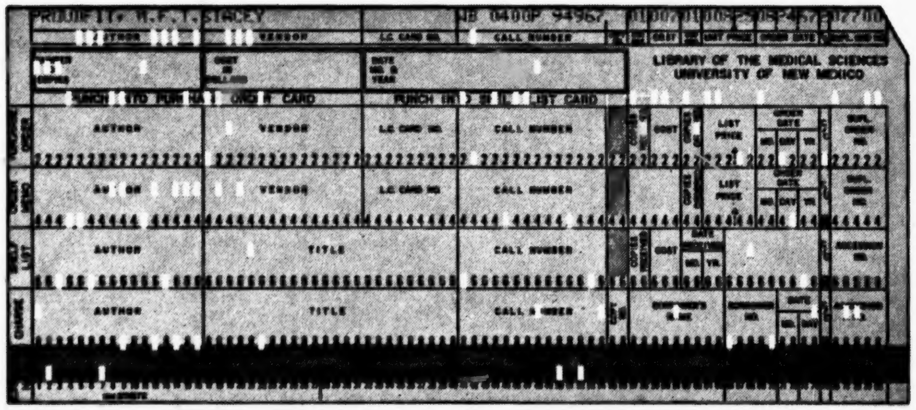

LIBRARY OF THE MEDICAL SCIENCES
UNIVERSITY OF NEW MEXICO
AUTHOR
VENDOR
LC CARD NO.
CALL NUMBER
TITLE

Figure 13

Purchase Order Card

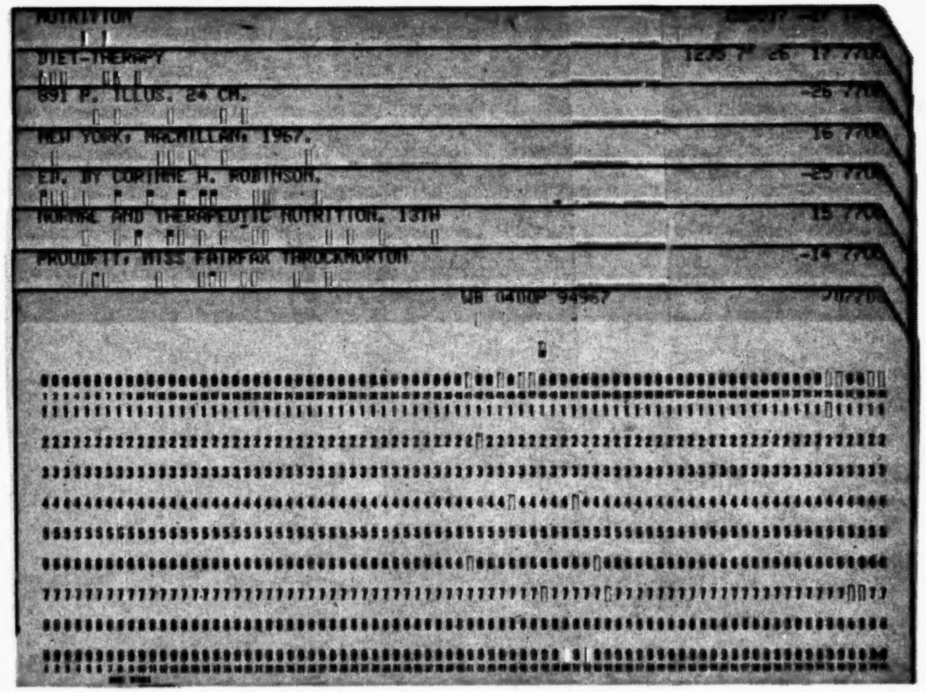

NUTRITION
DIET-THERAPY
891 P. ILLUS. 24 CM.
NEW YORK, MACMILLAN, 1967.
ED. BY CORINNE H. ROBINSON.
NORMAL AND THERAPEUTIC NUTRITION. 13TH
PROUDFIT, MISS FAIRFAX THROCKMORTON
QB 0400P 94967

Figure 14

Catalog Cards

APPENDIX B

2260 SCREEN DISPLAYS

Order

1. BEGIN INQUIRY

   ORDER

2. PLEASE SUBMIT YOUR IDENTIFICATION NUMBER.

   2525707772

3. YOU ARE NOT A QUALIFIED USER-STOP-
   OR SUBMIT CORRECT IDENTIFICATION NUMBER.

   4525707773

4. ENTER THE TITLE TO BE ORDERED, NOT
   MORE THAN 240 CHARACTERS PLEASE.

   ANATOMY OF THE HUMAN BODY

5. ENTER THE EDITION NUMBER,
   3 CHARACTERS PLEASE.

   001

6. ENTER PLACE OF PUBLICATION NOT
   MORE THAN 120 CHARACTERS PLEASE.

   PHILADELPHIA

7. ENTER PUBLISHER NOT MORE THAN
   68 CHARACTERS PLEASE.

   LIPPINCOTT

8. ENTER DATE OF PUBLICATION

   1959

```
9.  ENTER ACCESSION NUMBER.

    B000580101


10. ENTER AUTHOR, LAST NAME FIRST
    AND NOT MORE THAN 213 CHARACTERS
    PLEASE.

    LOCKHART, G. E.


11. ENTER VENDOR NUMBER.
```

Receive

12. BEGIN INQUIRY

    RECEIVE

13. PLEASE SUBMIT YOUR IDENTIFICATION NUMBER.

    4525707773

14. PLEASE ENTER ACCESSION NO.-DROP OR
    RETAIN-DATE OF RECEIPT-ACTUAL COST
    IN ONE OF THE FOLLOWING FORMATS
    XXXXXXXXXX-MM/DD/YY-$$$.$$-RETAIN OR
    XXXXXXXXXX-DROP

    B000580101-03/15/69-010.00-RETAIN

15. PLEASE PASS ITEM B000580101 TO CATALOGING.

Catalog

```
16. BEGIN INQUIRY

    CATALOG


17. PLEASE SUBMIT YOUR IDENTIFICATION
    NUMBER.

    4525707773


18. DO YOU WISH TO UPDATE THE AUTHOR
    PLEASE ENTER YES-ACCESSION NUMBER
    OR NO IN THE FOLLOWING FORMAT
    YES-XXXXXXXXXX OR
    NO

    YES-B000580101


19. DO YOU WISH TO ADD OR DROP THE AUTHOR
    PLEASE ENTER ADD OR DROP.

    ADD


20. WHO IS THE AUTHOR.  PLEASE ENTER
    LAST-NAME, FIRST-NAME MIDDLE-NAME.

    HAMILTON, F. W.


21. THE APPROPRIATE RECORD HAS BEEN
    SUCCESSFULLY UPDATED.


22. DO YOU WISH TO UPDATE THE AUTHOR
    PLEASE ENTER YES-ACCESSION NUMBER
    OR NO IN THE FOLLOWING FORMAT
    YES-XXXXXXXXXX OR
    NO

    NO
```

23. DO YOU WISH TO UPDATE THE TITLE
    PLEASE ENTER YES-ACCESSION NUMBER
    OR NO IN THE FOLLOWING FORMAT
    YES-XXXXXXXXXX OR
    NO

    NO

24. DO YOU WISH TO UPDATE THE SUBJECT
    PLEASE ENTER YES-ACCESSION NUMBER
    OR NO IN THE FOLLOWING FORMAT
    YES-XXXXXXXXXX OR
    NO

    YES-B000580101

25. DO YOU WISH TO ADD OR DROP A SUBJECT
    PLEASE ENTER ADD OR DROP

    ADD

26. WHAT IS THE SUBJECT. PLEASE ENTER IT.

    HEART DEFECTS, CONGENITAL

27. THE APPROPRIATE RECORD HAS BEEN
    SUCCESSFULLY UPDATED.

28. DO YOU WISH TO UPDATE THE SUBJECT
    PLEASE ENTER YES-ACCESSION NUMBER
    OR NO IN THE FOLLOWING FORMAT
    YES-XXXXXXXXXX OR
    NO

    YES-B000580101

29. DO YOU WISH TO ADD OR DROP A SUBJECT
    PLEASE ENTER ADD OR DROP.

    ADD

30. WHAT IS THE SUBJECT. PLEASE ENTER IT.

    HEART SURGERY

```
31. THE APPROPRIATE RECORD HAS BEEN
    SUCCESSFULLY UPDATED.


32. DO YOU WISH TO UPDATE THE SUBJECT
    PLEASE ENTER YES-ACCESSION NUMBER
    OR NO IN THE FOLLOWING FORMAT
    YES-XXXXXXXXXX OR
    NO

    NO


33. DO YOU WISH TO UPDATE THE CALL NUMBER
    PLEASE ENTER YES-ACCESSION NUMBER
    OR NO IN THE FOLLOWING FORMAT
    YES-XXXXXXXXXX OR
    NO

    YES-B000580101


34. PLEASE ENTER THE CALL NUMBER.

    WC  440SM  6959


35. THE APPROPRIATE RECORD HAS BEEN
    SUCCESSFULLY UPDATED.


36. DO YOU WISH TO UPDATE THE NOTES
    PLEASE ENTER YES-ACCESSION NUMBER
    OR NO IN THE FOLLOWING FORMAT
    YES-XXXXXXXXXX OR
    NO

    YES-B000580101


37. PLEASE ENTER THE NOTES.

    AMERICAN LECTURE SERIES NO.267


38. THE APPROPRIATE RECORD HAS BEEN
    SUCCESSFULLY UPDATED.
```

```
39. DO YOU WISH TO UPDATE THE COLLATION
    PLEASE ENTER YES-ACCESSION NUMBER
    OR NO IN THE FOLLOWING FORMAT
    YES-XXXXXXXXXX OR
    NO

    YES-B000580101

40. PLEASE ENTER THE COLLATION.

    276 P. ILLUS. 25CM.

41. DO YOU WISH TO CATALOG ANOTHER ITEM

    NO

42. BEGIN INQUIRY
```

Search one

```
43. BEGIN INQUIRY

    452507773


44. YOU MAY SEARCH BY AUTHOR, TITLE, SUBJECT,
    OR ANY COMBINATION THEREOF.
    ENTER YOUR CATEGORY FROM ONE OF THE FOLLOWING:
    A,T,S,AT,AS,TS,ATS.

    A


45. ENTER YOUR REQUESTED AUTHOR.  NOT MORE
    THAN 213 CHARACTERS PLEASE.

    BERENS, CONRAD


46. THERE ARE 4 CITATIONS RELATIVE TO YOUR
    REQUEST.  WOULD YOU CARE TO FURTHER
    DEFINE YOUR REQUEST WITH THE FOLLOWING OPTIONS:
    BOOKS ONLY....JOURNALS ONLY....
    LANGUAGE....DATE OF PUBLICATION YY TO YY.

    DATE 61-68


47. WW/0168/B  45/961
    BERENS, CONRAD
    ATLAS OF OPHTHALMIC SURGERY BY
    CONRAD BERENS AND JOHN HARRY KING, JR.
    DRAWINGS BY ROBERT BRAY WINGATE
    PHILADELPHIA, LIPPINCOTT, 1961.
    623 P. ILLUS. 29CM.
    EYE
    MEDICAL ILLUSTRATION
    KING, JOHN HARRY, JT. AUTH.


48. THERE ARE NO MORE CITATIONS RELATIVE
    TO YOUR REQUEST.

49. BEGIN INQUIRY
```

Search two

```
50. BEGIN INQUIRY

    4525628717


51. YOU MAY SEARCH BY AUTHOR, TITLE, SUBJECT
    OR ANY COMBINATION THEREOF.  ENTER YOUR
    CATAGORY FROM ONE OF THE FOLLOWING:
    A,T,S,ATS,AS,TS, OR AT.

    S


52. PLEASE SUBMIT YOUR REQUESTED SUBJECT,
    NOT MORE THAN 80 CHARACTERS.

    SURGERY


53. THERE ARE 29 CITATIONS RELATIVE TO
    SURGERY
    APPEARING IN THE FOLLOWING GENERAL AREAS
    SURGERY
    HEALTH OCCUPATIONS
    ENTER NAME OF GENERAL AREA OF INTEREST.

    HEALTH OCCUPATIONS


54. IS A MORE GENERAL OR A MORE SPECIFIC
    CATEGORY REQUIRED.

    MORE SPECIFIC


55. NEXT MORE SPECIFIC CATEGORIES ARE
    HEART SURGERY
    NEUROSURGERY
    ORTHOPEDICS
    PROCTOLOGY
    SURGERY, ORAL
    SURGERY, PLASTIC
    UROLOGY
    VASCULAR SURGERY
    DO YOU WISH TO REDEFINE YOUR SUBJECT.

    HEART SURGERY
```

```
56. WG/0017/R  61/957
    RODRIGUEZ, JORGE A.
    ATLAS OF CARDIAC SURGERY
    SAUNDERS
    PHILADELPHIA
    250 P. ILLUS. 30 CM.
    HEART SURGERY
    HEALTH OCCUPATIONS
    SPECIALTIES, MEDICAL
    SURGERY
    SURGERY, CARDIOVASCULAR

57. THERE ARE NO MORE CITATIONS ON FILE
    RELATIVE TO YOUR REQUEST.

58. BEGIN INQUIRY

    4525834645

59. YOU MAY SEARCH BY AUTHOR, TITLE, SUBJECT
    OR ANY COMBINATION THEREOF.  ENTER YOUR
    CATEGORY FROM ONE OF THE FOLLOWING:
    A,T,S,ATS,AS,TS, OR AT.

    S

60. PLEASE SUBMIT YOUR REQUESTED SUBJECT.
    NOT MORE THAN 80 CHARACTERS.

    HEAD

61. THERE ARE 5 CITATIONS RELATIVE TO
    HEAD
    APPEARING IN THE FOLLOWING GENERAL AREAS
    BODY REGIONS
    IS A MORE GENERAL OR MORE SPECIFIC
    CATEGORY REQUIRED

    MORE GENERAL

62. NEXT MORE GENERAL CATEGORY IS
    BODY REGIONS
    DO YOU WISH TO REDEFINE YOUR SUBJECT

    NO
```

63. WOULD YOU CARE TO FURTHER DEFINE YOUR
    SEARCH ACCORDING TO THE FOLLOWING OPTIONS
    BOOKS ONLY
    JOURNALS ONLY
    DATE OF PUBLICATION YY TO YY
    LANGUAGE

    NO

64. [ALL CITATIONS WOULD BE DISPLAYED AT THIS TIME]

Search three

```
65. BEGIN INQUIRY

    4525628717

66. YOU MAY SEARCH BY AUTHOR, TITLE, SUBJECT
    OR ANY COMBINATION THEREOF.  ENTER YOUR
    CATEGORY FROM ONE OF THE FOLLOWING:
    A,T,S,ATS,AS,TS, OR AT

    A

67. ENTER YOUR REQUESTED AUTHOR.  NOT MORE
    THAN 213 CHARACTERS PLEASE.

    LAUKONEN, SHIRLEY A.

68. THERE IS NO INFORMATION FOR YOUR REQUEST.
    TRY AGAIN.
```

APPENDIX C

SOURCE LISTINGS

PROGRAM NAME	PAGE
ACNFOR	198
ADDAUT	354
ADDSUB	368
ADDTIT	381
AUTFOR	198
AUTIO	174
CBLSECT	152
CLRSCX	333
DESCIO	189
DNTREE	320
DROPAUT	348
DROPSUBJ	362
DROPTITL	375
FLNMLST	202
IOENQ	161
IO2260	99
LOOKID	232
M1	388
M2	390
M3	394
M4	401
M5	404
M10	410
M11	204
M16	209
M17	216
M18	219
M19	222
M20	227
M50	230
M51	233
M52	235
M53	237
M54	239
M55	241
M56	246
M57	243
M60	248
M61	250
M70	255
M71	257
M72	259
M73	261
M74	264
M75	267
M76	269
M77	271
M78	273
M79	275
M81	278
M82	280
M83	282
M84	284
M85	287
M86	289
M87	291
M88	295
M89	302
M91	307
M92	309
M93	312
M94	314
M95	316
M99	319
OPNCLO	169
RDCALP	327
RDCNUM	329
RDPLAC	196
RDPUBL	197
RTLIST	167
RWACNU	194
RWACN1	331
RWAUTU	192
RWSB1U	192
RWSB2U	193
RWT1UP	193
RWT2UP	194
SCPTEXT	334
SEQAUT	172
SEQDSC	187
SEQSUB	182
SEQTIT	177
SUBIO	345
SUBJIO	184
SUBUPD	336
SUBWRT	341
SUBSRT	343
SU1FOR	199
SU2FOR	200
TERM1	155
TITLIO	179
TI1FOR	198
TI2FOR	199
TRTSTBL	150
UPTREE	322
WTACN	200
WTAUT	195
WTFORM	325
WTSUB1	199
WTSUB2	200
WTTI1	196
WTTI2	196

```
* 2260 I/O PACKAGE-MULT LINES WITH MULT TERMS ON EACH                   RSC10010
         TITLE 'BDU-ML 2260 PACKAGE'                                    RSC10020
IO2260  CSECT                                                           RSC10030
         ENTRY WRINIT                                                   RSC10040
         SPACE 5                                                        RSC10050
TCBLN    EQU   56                    LENGTH OF TCB IN BYTES             RSC10060
NOLNES   EQU   1                     NUMBER OF LINES                    RSC10070
NOTERM   EQU   2                     NUMBER OF TERMINALS                RSC10080
         SPACE 5                                                        RSC10090
* REGISTER DEFINITIONS                                                  RSC10100
* REGISTERS 12 & 4 ARE USED AS BASE REGISTERS                           RSC10110
LCBR     EQU   11                    LINE CONTROL BLOCK REGISTER        RSC10120
TCBR     EQU   10                    TERMINAL CONTROL BLOCK REGISTER    RSC10130
RETR     EQU   9                     BAL RETURN REGISTER                RSC10140
RETREG   EQU   9                     BAL RETURN REGISTER                RSC10150
DECBR    EQU   8                     DECB REGISTER                      RSC10160
SAVR1    EQU   1                     SAVE REGISTER 1                    RSC10170
SAVR2    EQU   2                     SAVE REGISTER 2                    RSC10180
DCBR     EQU   4                     DCB REGISTER                       RSC10190
WRKR1    EQU   3                     WORK REGISTERS 1 2 &3              RSC10200
WRKR2    EQU   14                    MUST  BE  EVEN                     RSC10210
WRKR3    EQU   WRKR2+1               MUST BE WRKR2 + 1                  RSC10220
MSGARR   EQU   7                     MSG AREA ADDR FOR I/O MACROS       RSC10230
LGTHR    EQU   6                     LENGTH REGISTER FOR I/O MACROS     RSC10240
TRMLR    EQU   5                     ADDR OF TRML LIST FOR MACROS       RSC10250
TABLR    EQU   3                     ADDRESS OF TRANSLATE TABLE         RSC10260
WRTREG   EQU   2                     ADDRESS OF WRITE ROUTINE           RSC10270
         SPACE 5                                                        RSC10280
         USING IECTDECB,DECBR        DECB DSECT                         RSC10290
         USING LCBDS,LCBR            LCB DSECT                          RSC10300
         USING TCBDS,TCBR            TCB DSECT                          RSC10310
         EJECT                                                          RSC10320
*********************************************************************** RSC10330
*********************************************************************** RSC10340
**                                                                   ** RSC10350
**                           I N I T                                 ** RSC10360
**                                                                   ** RSC10370
*********************************************************************** RSC10380
*********************************************************************** RSC10390
**                                                                   ** RSC10400
**  CALLED BY.   DROPL, DROPT, TCLA.                                 ** RSC10410
**  ROUTINES CALLED.   CLSRN(EXTERNAL), RETURNS TO O.S.              ** RSC10420
**                                                                   ** RSC10430
**  ABENDS. 4,5.                                                     ** RSC10440
**  CALLING CONVENTIONS.                                             ** RSC10450
**     NONE.                                                         ** RSC10460
**  FUNCTIONS.                                                       ** RSC10470
**     1. GENERAL HOUSEKEEPING.                                      ** RSC10480
**     2. OBTAIN AND SET THE TIMER INTERVAL & MAXIMUM NUMBER OF SUB- ** RSC10490
**        TASKS PARAMETERS.                                          ** RSC10500
**     3. SET END OF LIST INDICATORS IN THE SCHEDULER CONTROL BLOCK  ** RSC10510
**        (SCB),THE DECB WAIT LIST AND THE TCB WAIT LIST.            ** RSC10520
**     4. INITIALIZE TSTAT TO A X'02' FOR EACH TCB.                  ** RSC10530
**     5. CALL SUBROUTINE 'OPNRN' SO THAT USER DATA FILES CAN BE     ** RSC10540
**        OPENED.                                                    ** RSC10550
**     6. OPEN THE DCB FOR REMOTE 2260'S.                            ** RSC10560
**     7. ISSUE DUMMY WAITR TO SHIFT MFT SCHEDULER.                  ** RSC10570
**                                                                   ** RSC10580
*********************************************************************** RSC10580
         SPACE 5                                                        RSC10590
INIT     SAVE  (14,12),,*              SAVE REGISTERS                   RSC10600
         BALR  12,0                                                     RSC10610
         USING *,12                                                     RSC10620
         USING *+4095,4                                                 RSC10630
         LA    4,4095(12)                                               RSC10640
BASEX    ST    13,SA+4                 CALLING PROGRAM SAVE AREA        RSC10650
         LR    11,13                   INSURE CALLING                   RSC10660
         LA    13,SA                     PROGRAM KNOWS WHERE            RSC10670
         ST    13,8(11)                  IT WENT                        RSC10680
         SPACE 5                                                        RSC10690
         L     WRKR1,0(1)              LOAD ADDR OF TIME-TASK IN EXEC   RSC10700
         CLC   0(2,WRKR1),=H'0'        ARE ANY PARAMETERS PRESENT       RSC10710
         BE    DEFAULT                 NO.TAKE DEFAULT OPTION           RSC10720
         CLC   0(2,WRKR1),=H'2'         IS JUST TIME PRESENT            RSC10730
         BE    PARAM1                   YES.                            RSC10740
         CLC   0(2,WRKR1),=H'5'         ARE BOTH PARAMETERS PRESENT     RSC10750
         BE    PARAM2                   YES.                            RSC10760
         LA    1,4(0)                   SET UP FOR ABEND                RSC10770
         B     ABEND                    BRANCH                          RSC10780
PARAM2   MVC   LARWKA1+6(2),5(WRKR1)    MOVE THE CHARACTERS             RSC10790
         PACK  LARWKA1,LARWKA1          PACK THEM                       RSC10800
         CVB   WRKR2,LARWKA1            CONVERT TO BINARY               RSC10810
         C     WRKR2,=F'16'             CHECK FOR TOO MANY TASKS        RSC10820
         BL    STM00100                                                 RSC10830
         LA    1,5(0)                                                   RSC10840
         B     ABEND                                                    RSC10850
STM00100 ST    WRKR2,PARM2              STORE                           RSC10860
PARAM1   MVC   LARWKA1,=D'0'            ZERO WORK AREA                  RSC10870
         MVC   LARWKA1+6(2),2(WRKR1)    MOVE THE CHARACTERS             RSC10880
         PACK  LARWKA1,LARWKA1          PACK THEM                       RSC10890
         CVB   WRKR3,LARWKA1            CONVERT TO BINARY               RSC10900
         M     WRKR2,=F'100'            ALLOW FOR SET TIMER ARGUMENT    RSC10910
         ST    WRKR3,PARM1              STORE                           RSC10920
         SPACE 5                                                        RSC10930
DEFAULT  LA    WRKR2,NOLNES-1(0)          LOAD NO. OF LINES - 1         RSC10940
         SLL   WRKR2,2(0)                 MULTIPLY BY 4                 RSC10950
         LA    WRKR3,SCB                                                RSC10960
         LA    WRKR3,4(WRKR2,WRKR3)                                     RSC10970
         MVI   0(WRKR3),X'80'             EOL IN SCB                    RSC10980
         LA    WRKR3,ELLX1                                              RSC10990
         LA    WRKR3,0(WRKR2,WRKR3)                                     RSC11000
         MVI   0(WRKR3),X'80'           EOL IN DECBWT LIST              RSC11010
         LA    WRKR1,NOTERM-1(0)                                        RSC11020
         SLL   WRKR1,2(0)                                               RSC11030
         LA    WRKR3,ELLX2                                              RSC11040
         LA    WRKR3,0(WRKR1,WRKR3)                                     RSC11050
         MVI   0(WRKR3),X'80'           EOL IN TCBWT LIST               RSC11060
         SPACE 5                                                        RSC11070
* NEED TO INITIALIZE IOS1 IN EACH TERMINAL CONTROL BLOCK                RSC11080
```

```
         LA    WRKR1,ALCB1                                              RSC11090
BTS0503  L     LCBR,0(WRKR1)                                            RSC11100
         L     DECBR,ADECB                                              RSC11110
         MVI   DECSDECB,X'40'                                           RSC11120
         LA    WRKR2,ATCB1                                              RSC11130
BTS0502  L     TCBR,0(WRKR2)            ADDR OF TCB                     RSC11140
         MVI   TSTAT,X'02'              TURN ON IOS1 BIT FOR EACH TERM  RSC11150
         TM    0(WRKR2),X'80'                                           RSC11160
         BO    BTS0501                  BR IF LAST TERM ON LINE         RSC11170
         LA    WRKR2,4(WRKR2)           OTHERWISE GET NEXT TCB          RSC11180
         B     BTS0502                                                  RSC11190
BTS0501  TM    0(WRKR1),X'80'                                           RSC11200
         BO    BTS0504                  BR IF NO MORE LINES             RSC11210
         LA    WRKR1,4(WRKR1)           NEXT LINE                       RSC11220
         B     BTS0503                                                  RSC11230
         SPACE 5                                                        RSC11240
* ISSUE OPEN TO PROBLEM PROGRAMS                                        RSC11250
BTS0504  CALL  OPNRN                                                    RSC11260
         SPACE 5                                                        RSC11270
* OPEN LINES VIA OPEN MACRO                                             RSC11280
         ST    4,SVBASE2                                                RSC11290
         L     DCBR,DCBADR                                              RSC11300
         OPEN  ((DCBR))                                                 RSC11310
         L     4,SVBASE2                                                RSC11320
         SPACE 5                                                        RSC11330
         B     SCHDULR                                                  RSC11360
         SPACE 5                                                        RSC11370
* STORAGE AREA                                                          RSC11380
SVBASE2  DC    F'0'                                                     RSC11390
SA       DS    18F                      SAVE AREA FOR LINKAGE AND REG   RSC11400
PARM1    DC    F'200'                   DEFAULT...SECONDS               RSC11410
PARM2    DC    F'1'                           ...SUBTASKS               RSC11420
LARWKA1  DC    D'0'                     FOR PACKING                     RSC11440
         EJECT                                                          RSC11450
*********************************************************************** RSC11460
*********************************************************************** RSC11470
```

```
**                                                                    ** RSC11480
**                          T E R M I N I A T                         ** RSC11490
**                                                                    ** RSC11500
************************************************************************ RSC11510
************************************************************************ RSC11520
**                                                                    ** RSC11530
**  CALLED BY.   O.S.                                                 ** RSC11540
**  ROUTINES CALLED.   OPNRN(EXTERNAL), SCHDULR.                      ** RSC11550
**  CALLING CONVENTIONS.                                              ** RSC11560
**     NONE.                                                          ** RSC11570
**  FUNCTIONS.                                                        ** RSC11580
**     1. WAIT FOR ALL TERMINALS TO FINISH MSG PROCESSING.            ** RSC11590
**     2. CALL SUBROUTINE 'CLSRN' TO ENABLE USER TO CLOSE DATA        ** RSC11600
**        FILES.                                                      ** RSC11610
**     3. CLOSE REMOTE 2260 DCB.                                      ** RSC11620
**     4. TERMINATE PROGRAM BY RETURNING TO O.S.                      ** RSC11630
**                                                                    ** RSC11640
************************************************************************ RSC11650
         SPACE 5                                                         RSC11660
TERMINAT LA    LCBR,ALCB1                                                RSC11670
         LR    WRKR1,LCBR               SAVE PTR TO ADDR OF CURRENT LCB RSC11680
BTS0107  L     LCBR,0(LCBR)                                              RSC11690
         TM    LNSTAT,X'01'                                              RSC11700
         BO    BTS0110                  BRANCH IF LINE DROPPED           RSC11710
         LA    TCBR,ATCB1                                                RSC11720
         LR    WRKR2,TCBR               SAVE PTR TO ADDR OF CURRENT TCB RSC11730
BTS0105  L     TCBR,0(WRKR2)                                             RSC11740
         TM    TSTAT,X'01'                                               RSC11750
         BO    BTS0106                  BR IF TERMINAL DROPPED           RSC11760
         TM    TECB,X'40'                                                RSC11770
         BZ    BTS0108                  BRANCH IF TERM IS MSG PRCSSING   RSC11780
BTS0106  TM    0(WRKR2),X'80'                                            RSC11790
         BO    BTS0110                  BR IF LAST TRM ON LINE           RSC11800
         LA    WRKR2,4(WRKR2)           GET NEXT PTR TO ADDR OF TCB      RSC11810
         B     BTS0105                                                   RSC11820
BTS0110  TM    0(WRKR1),X'80'                                            RSC11830
```

```
         BO    BTS0109                  BR IF NO MORE LINES              RSC11840
         LA    WRKR1,4(WRKR1)           GET NEXT LCB PTR                 RSC11850
         B     BTS0107                                                   RSC11860
BTS0108  WAIT  1,ECB=(TCBR)             WAIT UNTIL THIS TERM FINISHES    RSC11870
         B     BTS0106                                                   RSC11880
BTS0109  CALL  CLSRN                                                     RSC11890
         ST    4,SVBASE2                                                 RSC11900
         L     DCBR,DCBADR                                               RSC11910
         CLOSE ((DCBR))                                                  RSC11920
         L     4,SVBASE2                                                 RSC11930
         L     13,SA+4                                                   RSC11940
         RETURN (14,12)                                                  RSC11950
         EJECT                                                           RSC11960
************************************************************************ RSC11970
************************************************************************ RSC11980
**                                                                    ** RSC11990
**                              A D D L                               ** RSC12000
**                                                                    ** RSC12010
************************************************************************ RSC12020
************************************************************************ RSC12030
**                                                                    ** RSC12040
**  CALLED BY.   TCLA.                                                ** RSC12050
**  ROUTINES CALLED.   PRWTANS.                                       ** RSC12060
**  CALLING CONVENTIONS.                                              ** RSC12070
**     1. TCBR MUST POINT TO TCB OF TERMINAL ISSUING COMMAND.         ** RSC12080
**     2. MSGARR MUST PT TO SCPBF+4 OF TERMINAL ISSUING THE COMMAND.  ** RSC12090
**  FUNCTIONS.                                                        ** RSC12100
**     1. ADD ALL TERMINALS ON THE LINE SPECIFIED.                    ** RSC12110
**     2. SET UP PART OF REPLY MESSAGE FOR TERMINAL ISSUING TER-      ** RSC12120
**        MINAL COMMAND.                                              ** RSC12130
**                                                                    ** RSC12140
************************************************************************ RSC12150
         SPACE 5                                                         RSC12160
* /ADDL NM, IS THE REQUIRED MESSAGE FORMAT                               RSC12170
         SPACE 3                                                         RSC12180
ADDL     OI    TSTAT,X'02'              TURN ON IO STAT BIT              RSC12190
```

```
         MVC   NTSTA,6(MSGARR)         GET LINE NBR                     RSC12200
         BAL   RETR,VALINO             IS THE LINE NBR VALID            RSC12210
         ST    LCBR,LCBRSW             SAVE  LCBR,TCBR                  RSC12220
         ST    TCBR,TCBRSW                                              RSC12230
         BAL   RETR,GETLCB             GET LCB FOR LINE TOBE ADDED      RSC12240
         LR    LCBR,WRKR3                                               RSC12250
         TM    LNSTAT,X'01'            IS THE LINE DOWN                 RSC12260
         BZ    LNDMSG                  IF NO GOTO LINE NOT DOWN MSG     RSC12270
         MVI   LNSTAT,X'00'            IF YES BRING LINE UP             RSC12280
         LA    SAVR1,ATCB1             GET PTR TO FIRST TERM ON LINE    RSC12290
BRNGUPT  L     TCBR,0(SAVR1)           GET TCBR AND START BRINGGING UP  RSC12300
         MVI   TECB,X'40'              THE TERM- SET TECB TO NOT BUSY   RSC12310
         MVI   TSTAT,X'02'             TURN ON IO STAT BIT              RSC12320
         LA    WRKR1,WRINIT            GET ADDR OF WRINIT               RSC12330
         ST    WRKR1,NXTRN1            PUT ADDR OF WRINIT IN NXTRN1     RSC12340
         TM    0(SAVR1),X'80'          IS THIS THE LAST TERM ON LINE    RSC12350
         BO    LSTRMOL                 IF YES RETURN                    RSC12360
         LA    SAVR1,4(SAVR1)          IF NO UPDATE PTR TO NXT TCB      RSC12370
         B     BRNGUPT                 GO BRING UP NEXT TERM            RSC12380
LSTRMOL  L     LCBR,LCBRSW             RESTORE LCBR,TCBR                RSC12390
         L     TCBR,TCBRSW                                              RSC12400
         B     PRWTANS5                                                 RSC12410
LNDMSG   MVC   0(22,MSGARR),=C'THE LINE IS ALREADY UP'                  RSC12420
         MVI   22(MSGARR),X'03'                                         RSC12430
         LA    LGTHR,22(0)                                              RSC12440
DBLUP    L     LCBR,LCBRSW                                              RSC12450
         L     TCBR,TCBRSW                                              RSC12460
         B     PRWTANS3                                                 RSC12470
         EJECT                                                          RSC12480
*********************************************************************** RSC12490
*********************************************************************** RSC12500
**                                                                   ** RSC12510
**                            A D D T                                ** RSC12520
**                                                                   ** RSC12530
*********************************************************************** RSC12540
*********************************************************************** RSC12550
```

```
**                                                                   ** RSC12560
**  CALLED BY.   TCLA.                                               ** RSC12570
**  ROUTINES CALLED.   ROUTINES IN 'ADDL', PRWTANS.                  ** RSC12580
**  CALLING CONVENTIONS.                                             ** RSC12590
**      1. TCBR MUST POINT TO TCB OF TERMINAL ISSUING COMMAND.       ** RSC12600
**      2. MSGARR MUST PT TO SCPBF+4 OF TERMINAL ISSUING THE COMMAND. ** RSC12610
**  FUNCTIONS.                                                       ** RSC12620
**      1. ADD TERMINAL SPECIFIED.                                   ** RSC12630
**      2. SET UP PART OF REPLY MESSAGE FOR TERMINAL ISSUING TERMINAL ** RSC12640
**         COMMAND.                                                  ** RSC12650
**                                                                   ** RSC12660
*********************************************************************** RSC12670
         SPACE 5                                                        RSC12680
* /ADDT NM. IS THE REQUIRED FORMAT OF ADD TERM MESSAGE                  RSC12690
ADDT     ST    LCBR,LCBRSW            SAVE LCBR,TCBR                    RSC12700
         ST    TCBR,TCBRSW                                              RSC12710
         MVI   TSTAT,X'02'                                              RSC12720
         MVC   NTSTA,6(MSGARR)        GET TERM NBR TO BE ADDED          RSC12730
         BAL   RETR,VALNBR            GO CHECK FOR VALID TERM NBR       RSC12740
         BAL   RETR,GETCB             GET TCB FOR THIS TERM             RSC12750
         L     LCBR,TLCB              GET LCB FOR THIS TERM             RSC12760
         TM    LNSTAT,X'01'           IS THE LINE DOWN                  RSC12770
         BO    ADDANS3                GO SET UP MSG                     RSC12780
         TM    TSTAT,X'01'            IS THE TERM DROPPED               RSC12790
         BZ    ADDANS4                IF NO GO TO ANS 4                 RSC12800
         MVI   TECB,X'40'                                               RSC12810
         LA    WRKR1,WRINIT           GET ADD OF WRINIT                 RSC12820
         ST    WRKR1,NXTRN1           PUT ADD OF WRINIT IN NXT RTN 1    RSC12830
         MVI   TSTAT,X'02'            TURN ON IO STATUS BIT             RSC12840
ADDANS4  L     LCBR,LCBRSW            RESTORE LCBR,TCBR                 RSC12850
         L     TCBR,TCBRSW                                              RSC12860
         B     PRWTANS4               GO PREPARE TO WRT ANS 4           RSC12870
ADDANS3  MVC  0(40,MSGARR),=C'TERMINAL CAN NOT BE ADDED. LINE IS DOWN.' RSC12880
         MVI   40(MSGARR),X'03'                                         RSC12890
         LA    LGTHR,40(0)            PUT LENGTH OF MSG IN LENGTH REG   RSC12900
         B     DBLUP                                                    RSC12910
         SPACE 5                                                        RSC12920
GETCB    MVC   LARWKA1,=D'0'           ZERO WORK AREA                   RSC12930
         MVC   LARWKA1+6(2),NTSTA       GET NM, THE PHY TERM NBR        RSC12940
         PACK  LARWKA1,LARWKA1          PACK PHY TERM NBR               RSC12950
         CVB   WRKR1,LARWKA1           CONVERT NM TO BINARY             RSC12960
         S     WRKR1,=F'1'             SUB 1 FROM PHY TERM NBR          RSC12970
         LA    WRKR3,TCBLN(0)          PUT LNGTH OF TCB IN REG          RSC12980
         MR    WRKR2,WRKR1             WRKR3 IS NOW INDEX TO RIGHT TCB  RSC12990
         L     WRKR1,ALCB1              GET ADDR OF FIRSTLCB            RSC13000
         L     WRKR1,8(WRKR1)           ADDR OF FIRST TCB               RSC13010
         LA    TCBR,0(WRKR1,WRKR3)     GET ADDR OF TCB TO BE DROPPED    RSC13020
         BR    RETR                                                     RSC13030
         SPACE 5                                                        RSC13040
VALNBR   SR    WRKR1,WRKR1             ZERO ERROR RETURN REG            RSC13050
         CLI   NTSTA,X'F0'              IS N LT MIN                     RSC13060
         BL    VALERR                                                   RSC13070
         CLI   NTSTA,X'F0'              IS N GT MAX                     RSC13080
         BH    VALERR                                                   RSC13090
         CLI   NTSTA+1,X'F1'            IS M LT MIN                     RSC13100
         BL    VALERR                                                   RSC13110
         CLI   NTSTA+1,X'F2'            IS M GT MAX                     RSC13120
         BH    VALERR                                                   RSC13130
         BR    RETR                                                     RSC13140
VALERR   MVC   0(24,MSGARR),=C'INVALID TERMINAL NUMBER.'                RSC13150
         MVI   24(MSGARR),X'03'                                         RSC13160
         LA    LGTHR,24(0)                                              RSC13170
         B     DBLUP                                                    RSC13180
NTSTA    DC    CL2'  '                 NUMBER TESTING AREA              RSC13190
         EJECT                                                          RSC13200
*********************************************************************** RSC13210
*********************************************************************** RSC13220
**                                                                   ** RSC13230
**                         D E C B W T                               ** RSC13240
**                                                                   ** RSC13250
*********************************************************************** RSC13260
*********************************************************************** RSC13270
```

```
**
**  CALLED BY.   SCHDULR.                                              ** RSC13280
**  ROUTINES CALLED.   SCHDULR.                                        ** RSC13290
**  CALLING CONVENTIONS.                                               ** RSC13300
**     1. AT LEAST ONE DECB MUST ALREADY HAVE THE COMPLETE BIT ON      ** RSC13310
**        OR WILL HAVE IT TURNED ON BY O.S. WHEN AN I/O OPERATION      ** RSC13320
**        ON THE LINE IS COMPLETED.                                    ** RSC13330
**  FUNCTIONS.                                                         ** RSC13340
**     1. WAIT UNTIL ONE LINE HAS ITS DECB POSTED AS HAVING COM-       ** RSC13350
**        PLETED AN I/O OPERATION.                                     ** RSC13360
**     2. UPDATE NXTLCB IN SCB TO CONTAIN THE POINTER TO THE LCB       ** RSC13370
**        THAT COMPLETED THE I/O OPERATION.                            ** RSC13380
**                                                                     ** RSC13390
**                                                                     ** RSC13400
*********************************************************************** RSC13410
         SPACE 5                                                        RSC13420
DECBWT   TWAIT (DECBR),ECBLIST=ELLX1      WAIT FOR ANY EVT TO COMPLETE  RSC13430
         L     LCBR,LADECB                ADDR OF LCB                   RSC13440
         LA    LCBR,0(LCBR)               ZERO HIGH ORDER BYTE          RSC13450
         LA    WRKR1,ALCB1                PT TO ADDR OF LCB1            RSC13460
STM00020 L     WRKR2,0(WRKR1)             ADDR OF LCB                   RSC13470
         LA    WRKR2,0(WRKR2)             ZERO HIGH ORDER BYTE          RSC13480
         CR    LCBR,WRKR2                                               RSC13490
         BE    STM00021                                                 RSC13500
         LA    WRKR1,4(WRKR1)                                           RSC13510
         B     STM00020                                                 RSC13520
STM00021 ST    WRKR1,NXTLCB                                             RSC13530
         B     SCHDULR                                                  RSC13540
ELLX1    DS    0F                                                       RSC13550
         DC    V(DECB1)                                                 RSC13560
         EJECT                                                          RSC13570
*********************************************************************** RSC13580
*********************************************************************** RSC13590
**                                                                   ** RSC13600
**                          D I S P L A Y                            ** RSC13610
**                                                                   ** RSC13620
*********************************************************************** RSC13630
```

```
*********************************************************************** RSC13640
**                                                                   ** RSC13650
**                                                                   ** RSC13660
**   CALLED BY.   TCLA.                                              ** RSC13670
**   ROUTINES CALLED.   PRWTANS.                                     ** RSC13680
**   CALLING CONVENTIONS.                                            ** RSC13690
**       1. TCBR MUST POINT TO TCB OF TERMINAL ISSUING COMMAND.      ** RSC13700
**       2. MSGARR MUST PT TO SCPBF+4 OF TERMINAL ISSUING THE COMMAND. ** RSC13710
**   FUNCTIONS.                                                      ** RSC13720
**       1. SET UP REPLY TO /DISPLAY COMMAND.                        ** RSC13730
**                                                                   ** RSC13740
*********************************************************************** RSC13750
         SPACE 5                                                        RSC13760
* DISPLAY ROUTINE...ENABLES A REQUESTOR TO INQUIRE WHICH TERMINALS      RSC13770
* (AND LINES) ARE IN AN 'ON' STATUS                                     RSC13780
* FIRST SET UP A LINE OF HEADING TO APPEAR AT TOP                       RSC13790
DISPLAY  MVC   0(17,MSGARR),=C'LINE TERMINALS UP'                       RSC13800
         MVI   17(MSGARR),X'0A'         INSERT NEW LINE SYMBOL          RSC13810
         ST    MSGARR,MSGARSAV          SAVE ORIG. MSG AREA             RSC13820
         LA    MSGARR,18(MSGARR)        UPDATE POINTER                  RSC13830
         OI    TSTAT,X'02'                                              RSC13840
         ST    TCBR,TCBRSW              SAVE TCBR                       RSC13850
         L     WRKR1,ALCB1              GET ADDR OF FIRSTLCB            RSC13860
         L     TCBR,8(WRKR1)            ADDR OF FIRST TCB               RSC13870
STM02000 SR    WRKR1,WRKR1                                              RSC13880
         IC    WRKR1,TLNO               LOAD BINARY NUMBER              RSC13890
         STC   WRKR1,LSAVX              SAVE LINE NO. FOR CHECKING      RSC13900
         LA    WRKR1,1(WRKR1)           INCREMENT BY 1                  RSC13910
         CVD   WRKR1,LARWKA1            CNVT TO DECIMAL                 RSC13920
         UNPK  0(2,MSGARR),LARWKA1+6(2) UNPACK TO OUTPUT AREA           RSC13930
         OI    1(MSGARR),X'F0'                                          RSC13950
         LA    MSGARR,4(MSGARR)         INCREMENT POINTER               RSC13940
         MVC   2(2,MSGARR),=C'  '                                       RSC13960
STM02004 TM    TSTAT,X'01'              IS TRM UP                       RSC13970
         BO    STM02001                 NO BRANCH                       RSC13980
         SR    WRKR1,WRKR1                                              RSC13990
         IC    WRKR1,TNO                YES LOAD BINARY NUMBER
         LA    WRKR1,1(WRKR1)          INCREMENT BY 1                   RSC14000
         CVD   WRKR1,LARWKA1           CNVT TO DECIMAL                  RSC14010
         UNPK  0(2,MSGARR),LARWKA1+6(2) UNPACK TO OUTPUT AREA           RSC14020
         OI    1(MSGARR),X'F0'                                          RSC14030
         LA    MSGARR,2(MSGARR)        INCREMENT POINTER                RSC14040
STM02001 LA    TCBR,TCBLN(TCBR)        UPDATE TCB                       RSC14050
         CLC   0(4,TCBR),=X'FFFFFFFF'  IS THE LAST TRM                  RSC14060
         BE    STM02003                IF YES PREPARE TO LEAVE          RSC14070
         CLC   TLNO(1),LSAVX           COMPARE                          RSC14080
         BE    STM02004                                                 RSC14090
         MVI   0(MSGARR),X'0A'         INSERT NEW LINE SYMBOL           RSC14100
         LA    MSGARR,1(MSGARR)                                         RSC14110
         MVC   LSAVX(1),TLNO                                            RSC14120
         B     STM02000                                                 RSC14130
STM02003 L     WRKR1,MSGARSAV          LOAD                             RSC14140
         LR    SAVR1,MSGARR                                             RSC14150
         SR    SAVR1,WRKR1             GET DIFFERENCE                   RSC14160
         LR    LGTHR,SAVR1                                              RSC14170
         MVI   0(MSGARR),X'03'         MOVE IN ETX                      RSC14180
         L     MSGARR,MSGARSAV                                          RSC14190
         L     TCBR,TCBRSW                                              RSC14200
         B     PRWTANS3                                                 RSC14210
LSAVX    DC    X'00'                                                    RSC14220
MSGARSAV DC    F'0'                                                     RSC14230
         EJECT                                                          RSC14240
************************************************************************RSC14250
************************************************************************RSC14260
**                                                                    **RSC14270
**                           D R O P L                                **RSC14280
**                                                                    **RSC14290
************************************************************************RSC14300
************************************************************************RSC14310
**                                                                    **RSC14320
**  CALLED BY.    TCLA.                                               **RSC14330
**  ROUTINES CALLED.   PRWTANS, TERMINAT.                             **RSC14340
**  CALLING CONVENTIONS.                                              **RSC14350
**      1. TCBR MUST POINT TO TCB OF TERMINAL ISSUING COMMAND.        ** RSC14360
**      2. MSGARR MUST PT TO SCPBF+4 OF TERMINAL ISSUING THE COMMAND. ** RSC14370
**   FUNCTIONS.                                                        ** RSC14380
**      1. DROP ALL TERMINALS ON LINE SPECIFIED.                       ** RSC14390
**      2. SET UP PART OF REPLY MESSAGE FOR TERMINAL ISSUING TERM-     ** RSC14400
**         INAL COMMAND.                                               ** RSC14410
**                                                                     ** RSC14420
*********************************************************************** RSC14430
         SPACE 5                                                        RSC14440
DROPL    OI    TSTAT,X'02'                                              RSC14450
         MVC   NTSTA,7(MSGARR)          STORE LINE NO.                  RSC14460
         BAL   RETR,VALINO                                              RSC14470
         BAL   RETR,GETLCB                                              RSC14480
         ST    LCBR,LCBRSW              A(LCB) IS IN WRKR3              RSC14490
         ST    TCBR,TCBRSW                                              RSC14500
         LR    LCBR,WRKR3                                               RSC14510
         TM    LNSTAT,X'01'             BR IF LINE DOWN                 RSC14520
         BO    BTS0401                                                  RSC14530
         OI    LNSTAT,X'01'             TURN ON LINE DOWN BIT           RSC14540
         LA    WRKR1,ATCB1                                              RSC14550
BTS0402  L     TCBR,0(WRKR1)                                            RSC14560
         MVI   TSTAT,X'01'             TURN ON TERM DOWN BIT            RSC14570
         MVC   TPCW,=F'1'               TURN ON /STOP-DROPPED BIT       RSC14580
         MVI   TECB,X'00'               ZERO ECB                        RSC14590
         TM    0(WRKR1),X'80'                                           RSC14600
         BO    BTS0403                  BR IF LAST TERM IN LCB          RSC14610
         LA    WRKR1,4(WRKR1)                                           RSC14620
         B     BTS0402                                                  RSC14630
BTS0403  LA    WRKR1,ALCB1                                              RSC14640
BTS0404  L     LCBR,0(WRKR1)                                            RSC14650
         TM    LNSTAT,X'01'                                             RSC14660
         BZ    BTS0401                  BR IF LINE NOT DOWN             RSC14670
         TM    0(WRKR1),X'80'                                           RSC14680
         BO    TERMINAT                                                 RSC14690
         LA    WRKR1,4(WRKR1)                                           RSC14700
         B     BTS0404                                                  RSC14710
```

```
BTS0401  L     LCBR,LCBRSW                                             RSC14720
         L     TCBR,TCBRSW                                             RSC14730
         B     PRWTANS2                                                RSC14740
         SPACE 5                                                       RSC14750
LNOERR   MVC   0(19,MSGARR),=C'INVALID LINE NUMBER'                    RSC14760
         LA    LGTHR,19(0)                                             RSC14770
         MVI   19(MSGARR),X'03'                                        RSC14780
         B     PRWTANS3                                                RSC14790
         SPACE 5                                                       RSC14800
VALINO   CLI   NTSTA,X'F0'             LOWER LIMIT OF HIGH ORDER DIGIT RSC14810
         BL    LNOERR                                                  RSC14820
         CLI   NTSTA,X'F0'             UPPER LIMIT OF HIGH ORDER DIGIT RSC14830
         BH    LNOERR                                                  RSC14840
         CLI   NTSTA+1,X'F1'           LOWER LIMIT OF LOW ORDER DIGIT  RSC14850
         BL    LNOERR                                                  RSC14860
         CLI   NTSTA+1,X'F1'           UPPER LIMIT OF LOW ORDER DIGIT  RSC14870
         BH    LNOERR                                                  RSC14880
         BR    RETR                                                    RSC14890
         SPACE 5                                                       RSC14900
GETLCB   MVC   LARWKA1,=D'0'          CONVERT LINE NO TO BINARY        RSC14910
         MVC   LARWKA1+6(2),NTSTA                                      RSC14920
         PACK  LARWKA1,LARWKA1                                         RSC14930
         CVB   WRKR1,LARWKA1                                           RSC14940
         S     WRKR1,=F'1'                                             RSC14950
         LA    WRKR2,ALCB1                                             RSC14960
         SLL   WRKR1,2(0)                                              RSC14970
         L     WRKR3,0(WRKR1,WRKR2)                                    RSC14980
         BR    RETR                                                    RSC14990
         EJECT                                                         RSC15000
*********************************************************************** RSC15010
*********************************************************************** RSC15020
**                                                                   ** RSC15030
**                           D R O P T                               ** RSC15040
**                                                                   ** RSC15050
*********************************************************************** RSC15060
*********************************************************************** RSC15070
```

```
**                                                                      ** RSC15080
**  CALLED BY.    ERRCVY, TCLA.                                         ** RSC15090
**  ROUTINES CALLED.   ROUTINES IN DROPL, PRWTANS, TERMINAT.            ** RSC15100
**  CALLING CONVENTIONS.                                                ** RSC15110
**      1. TCBR MUST POINT TO TCB OF TERMINAL ISSUING COMMAND.          ** RSC15120
**      2. MSGARR MUST PT TO SCPBF+4 OF TERMINAL ISSUING THE COMMAND.   ** RSC15130
**  FUNCTIONS.                                                          ** RSC15140
**      1. DROP TERMINAL SPECIFIED.                                     ** RSC15150
**      2. SET UP PART OF REPLY MESSAGE FOR TERMINAL SPECIFYING THE     ** RSC15160
**         TERMINAL COMMAND.                                            ** RSC15170
**                                                                      ** RSC15180
************************************************************************ RSC15190
         SPACE 5                                                           RSC15200
* /DROPT NM, IS THE REQUIRED MESSAGE FORMAT                                RSC15210
DROPT    ST    LCBR,LCBRSW                 SAVE LCBR AND TCBR              RSC15220
         ST    TCBR,TCBRSW                                                 RSC15230
         OI    TSTAT,X'02'                                                 RSC15240
         MVC   NTSTA(2),7(MSGARR)          MOVE TERM NBR TO TEST AREA      RSC15250
         BAL   RETR,VALNBR                 IS THE NBR A VALID NBR          RSC15260
         BAL   RETR,GETCB                                                  RSC15270
         L     LCBR,TLCB                   LOAD LCBR WITH LINE TO BE TESTD RSC15280
         TM    LNSTAT,X'01'                IS LINE DOWN                    RSC15290
         BO    DRPWA2                      IF YES GO GET READ TO WRT ANS 2 RSC15300
         TM    TSTAT,X'01'                 IS TERM DROPPED                 RSC15310
         BO    DRPWA1                      GO TO ARE ALL TERMS DROPPED     RSC15320
ERRDROP  MVC   TPCW,=F'1'                  TURN ON /STOP BIT OF TPCW       RSC15330
         MVC   TECB,=F'0'                  ZERO TECB                       RSC15340
         MVI   TSTAT,X'01'                 TURN ON BIT 7 OF TSTAT          RSC15350
DRAATD   LA    SAVR1,ATCB1                 GET PTR TO FIRST TCB ON LINE    RSC15360
DRITD    L     TCBR,0(SAVR1)               GET TCB                         RSC15370
         TM    TSTAT,X'01'                 IS TERM DROPPED                 RSC15380
         BZ    DRPWA1                      IF NO GET READY TO WRT ANS 1    RSC15390
         TM    0(SAVR1),X'80'              IS THIS THE LAST TCB            RSC15400
         BO    DRAALD                      IF YES GO TO ARE ALL LINES DOWN RSC15410
         LA    SAVR1,4(SAVR1)              IF NO GO UPDATE TCB PTR         RSC15420
         B     DRITD                       GO TO IS TERM DOWN              RSC15430
```

```
DRAALD   MVI   LNSTAT,X'01'                  TURN ON LINE DROPPED BIT         RSC15440
         LA    SAVR1,ALCB1                   GET FIRST LCB PTR                RSC15450
DRGTNT   L     LCBR,0(SAVR1)                 GET FIRST LCB                    RSC15460
         TM    LNSTAT,X'01'                  IS THE LINE DOWN                 RSC15470
         BZ    DRPWA2                        IF NO GET READY FOR ANS 2        RSC15480
         TM    0(SAVR1),X'80'                IS THIS THE LAST LCB             RSC15490
         BO    TERMINAT                      IF YES ALL LINES WERE DOWN       RSC15500
         LA    SAVR1,4(SAVR1)                UPDATE LCB PTR                   RSC15510
         B     DRGTNT                        GO TEST NEXT LCB                 RSC15520
DRPWA1   L     LCBR,LCBRSW                   RESTORE LCBR,TCBR                RSC15530
         L     TCBR,TCBRSW                                                    RSC15540
         B     PRWTANS1                      GO WRITE ANS1                    RSC15550
DRPWA2   L     LCBR,LCBRSW                   RESTORE LCBR,TCBR                RSC15560
         L     TCBR,TCBRSW                                                    RSC15570
         B     PRWTANS2                      GO WRITE ANS 2                   RSC15580
         EJECT                                                                RSC15590
************************************************************************RSC15600
************************************************************************RSC15610
**                                                                    **RSC15620
**                             E R R C K                              **RSC15630
**                                                                    **RSC15640
************************************************************************RSC15650
************************************************************************RSC15660
**                                                                    **RSC15670
**  CALLED BY.   SCHDULR.                                             **RSC15680
**  ROUTINES CALLED.   ERRC[illegible]Y, INFRMT, SCHDULR.                        **RSC15690
**  CALLING CONVENTIONS.                                              **RSC15700
**     1. TCBR MUST POINT TO TCB OF TERMINAL WHICH IS TO HAVE ITS     **RSC15710
**        LAST I/O OPERATION CHECKED.                                 **RSC15720
**     2. ERROR INFORMATION PLACED IN THE DECB BY BTAM AFTER COMPLE-  **RSC15730
**        TION OF THE I/O OPERATION FOR THIS TERMINAL MUST HAVE       **RSC15740
**        BEEN PREVIOUSLY STORED IN THE APPROPRIATE PLACES OF THE     **RSC15750
**        TCB.                                                        **RSC15760
**  FUNCTIONS.                                                        **RSC15770
**     1. CHECK ALL I/O OPERATIONS FOR ERRORS.                        **RSC15780
**     2. PASS CONTROL TO INFRMT FOR ALL READ OPERATIONS.             **RSC15790
**      3. WAIT FOR AN INTERVAL OF TIME SPECIFIED IN THE PARM OPER-    ** RSC15800
**         AND OF THE '// EXEC' CARD WHEN THERE IS NO TERMINAL ACT-    ** RSC15810
**         IVITY.                                                      ** RSC15820
**                                                                     ** RSC15830
************************************************************************ RSC15840
************************************************************************ RSC15850
         SPACE 5                                                         RSC15860
ERRCK    CLI   TSECB,X'7F'                                               RSC15870
         BNE   BTS0201                BRANCH IF I/O ERROR                RSC15880
         TM    TSTSFLGS,X'04'                                            RSC15890
         BO    BTS0202                BRANCH IF NEGATIVE RESPONSE        RSC15900
         SR    WRKR1,WRKR1           TURN OFF NEG RSPNS INDICATOR        RSC15910
         IC    WRKR1,TNO                                                 RSC15920
         LA    WRKR2,NRI(WRKR1)                                          RSC15930
         MVI   0(WRKR2),X'00'                                            RSC15940
         CLI   TOPTYP+1,X'81'                                            RSC15950
         BE    BTS0203                BRANCH IF OPTYP WAS A READ TI      RSC15960
         BAL   RETR,UDNXTRN           UPDATE NXTRN1 & IOS1 WITH NXTR2    RSC15970
         TM    TPCW+3,X'02'                                              RSC15980
         BO    INFRMT                 BRANCH IF READ FULL BUFFER         RSC15990
         B     SCHDULR                                                   RSC16000
         SPACE 3                                                         RSC16010
BTS0203  L     MSGARR,TSTAT           WAS THERE AN S/E WITH NO START     RSC16020
         CLI   6(MSGARR),X'03'        SYMBOL                             RSC16030
         BE    BTS0205                BRANCH IF 3RD CHAR IS ETX          RSC16040
         BAL   RETR,UDNXTRN           UPDATE NXTRN1 & IOS1               RSC16050
         TM    TPCW+3,X'04'                                              RSC16060
         BO    INFRMT                 BRANCH IF READ MI                  RSC16070
         B     SCHDULR                                                   RSC16080
         SPACE 3                                                         RSC16090
BTS0205  TM    TPCW+3,X'04'                                              RSC16100
         BO    BTS0206                BRANCH IF READ MI                  RSC16110
         LA    RETR,SCHDULR                                              RSC16120
         B     UDNXTRN                                                   RSC16130
BTS0206  LA    WRKR1,SPCWRT           A READ MI WAS TRIED WITHOUT A      RSC16140
         ST    WRKR1,NXTRN1           START SYMBOL. WRITE ERROR MSG      RSC16150
```

```
         B     SCHDULR                & TRY AGAIN.                       RSC16160
         SPACE 3                                                         RSC16170
BTS0201  LA    WRKR1,ERRCVY           SCHEDULE FOR ERROR RECOVERY        RSC16180
         ST    WRKR1,NXTRN1                                              RSC16190
         B     SCHDULR                                                   RSC16200
         SPACE 3                                                         RSC16210
BTS0202  SR    WRKR1,WRKR1           TURN ON BYTE IN NRI                 RSC16220
         IC    WRKR1,TNO                                                 RSC16230
         LA    WRKR2,NRI(WRKR1)                                          RSC16240
         MVI   0(WRKR2),X'FF'                                            RSC16250
         LA    WRKR1,SPCPOLL                                             RSC16260
         ST    WRKR1,NXTRN1                                              RSC16270
         SR    WRKR3,WRKR3                                               RSC16280
         LA    WRKR1,ALCB1                                               RSC16290
BTS0210  L     LCBR,0(WRKR1)                                             RSC16300
         TM    LNSTAT,X'01'                                              RSC16310
         BO    BTS0213                BR IF LINE DOWN                    RSC16320
         LA    WRKR2,ATCB1                                               RSC16330
BTS0212  L     TCBR,0(WRKR2)                                             RSC16340
         TM    TSTAT,X'01'                                               RSC16350
         BO    BTS0211                BR IF NEG RESPNS IND IN            RSC16360
         IC    WRKR3,TNO                                                 RSC16370
         LA    SAVR1,NRI(WRKR3)                                          RSC16380
         TM    0(SAVR1),X'FF'                                            RSC16390
         BO    BTS0211                BR IF NEG RESPNS IND ON            RSC16400
         TM    TECB,X'40'                                                RSC16410
         BO    SCHDULR                 BR IF NOT MSG PROCESSING          RSC16420
BTS0211  TM    0(WRKR2),X'80'                                            RSC16430
         BO    BTS0213                BR TO GET NEXT LINE                RSC16440
         LA    WRKR2,4(WRKR2)         GET NEXT TCB                       RSC16450
         B     BTS0212                                                   RSC16460
BTS0213  TM    0(WRKR1),X'80'                                            RSC16470
         BO    BTS0214                BR IF LAST LINE & SETIMER          RSC16480
         LA    WRKR1,4(WRKR1)                                            RSC16490
         B     BTS0210                                                   RSC16500
BTS0214  STIMER WAIT,BINTVL=PARM1                                        RSC16510
```

```
         MVC   NRI(16),=16X'00'         RESET NEG RESPNS INDS           RSC16520
         B     SCHDULR                                                  RSC16530
         SPACE 3                                                        RSC16540
UDNXTRN  MVC   NXTRN1,NXTRN2                                            RSC16550
         TM    BIOS2,X'02'                                              RSC16560
         BO    BTS0204                  BRANCH IF IOS2 ON               RSC16570
         NI    TSTAT,X'FD'              TURN OFF IOS1                   RSC16580
         BR    RETR                                                     RSC16590
BTS0204  OI    TSTAT,X'02'              TURN ON IOS1                    RSC16600
         BR    RETR                                                     RSC16610
         EJECT                                                          RSC16620
************************************************************************RSC16630
************************************************************************RSC16640
**                                                                    **RSC16650
**                            E R R C V Y                             **RSC16660
**                                                                    **RSC16670
************************************************************************RSC16680
************************************************************************RSC16690
**                                                                    **RSC16700
**  CALLED BY.   SCHDULR.                                             **RSC16710
**  ROUTINES CALLED.   DROPT.                                         **RSC16720
**  CALLING CONVENTIONS.                                              **RSC16730
**     1. TCBR MUST POINT TO TCB OF TERMINAL ISSUING COMMAND.         **RSC16740
**  FUNCTIONS.                                                        **RSC16750
**     1. PREPARE TO DROP THE TERMINAL BECAUSE OF I/O ERRORS.         **RSC16760
**                                                                    **RSC16770
************************************************************************RSC16780
         SPACE 5                                                        RSC16790
ERRCVY   L     MSGARR,TSTAT                                             RSC16800
         LA    MSGARR,4(MSGARR)                                         RSC16810
         ST    LCBR,LCBRSW              SAVE LCBR & TCBR        7/11/68 RSC16812
         ST    TCBR,TCBRSW                                      7/11/68 RSC16814
         B     ERRDROP                                                  RSC16820
         EJECT                                                          RSC16830
************************************************************************RSC16840
************************************************************************RSC16850
```

```
**                                                                    **RSC16860
**                          E W R I T E F B                           **RSC16870
**                                                                    **RSC16880
************************************************************************RSC16890
************************************************************************RSC16900
**                                                                    **RSC16910
**  CALLED BY.   SCHDULR.                                             **RSC16920
**  ROUTINES CALLED.   OUTFRMT.                                       **RSC16930
**  CALLING CONVENTIONS.                                              **RSC16940
**     1. TCBR MUST POINT TO TCB OF TERMINAL ABOUT TO PERFORM A       **RSC16950
**        WRITE.                                                      **RSC16960
**  FUNCTIONS.                                                        **RSC16970
**     1. LOADS MSGARR WITH ADDRESS OF I/O AREA, LGTHR WITH LENGTH    **RSC16980
**        OF OUTPUT MESSAGE SPECIFIED BY USER, & TRMLR WITH ADDRESS   **RSC16990
**        OF THE TERMINAL LIST IN PREPERATION FOR THE WRITE MACRO.    **RSC17000
**     2. ERRCK IS STORED IN NXTRN1, AND MSGPROC IN NXTRN2.           **RSC17010
**                                                                    **RSC17020
************************************************************************RSC17030
         SPACE 5                                                        RSC17040
EWRITEFB L     DECBR,ADECB              LOAD ADDR OF DECB               RSC17050
         L     MSGARR,TSTAT             LOAD ADDR OF MSG AREA           RSC17060
         LA    MSGARR,4(MSGARR)         ALLOW FOR COUNT                 RSC17070
         L     LGTHR,TML                LOAD LENGTH                     RSC17080
         L     TRMLR,TLST               LOAD ADDR OF TERMINAL LIST      RSC17090
         LA    WRKR1,ERRCK              PUT ADDR OF ERRCK IN NXTRN1     RSC17100
         ST    WRKR1,NXTRN1                                             RSC17110
         LA    WRKR1,MSGPROC            PUT ADDR OF MSG PROC IN NXTRN2  RSC17120
         ST    WRKR1,NXTRN2                                             RSC17130
         NI    BIOS2,X'FD'              TURN OFFBIOS2                   RSC17140
         LA    WRTREG,WRITE             LOAD WRITE REGISTER             RSC17150
         B     OUTFRMT                  BRANCH                          RSC17160
         EJECT                                                          RSC17170
************************************************************************RSC17180
************************************************************************RSC17190
**                                                                    **RSC17200
**                            I N F R M T                             **RSC17210
**                                                                    ** RSC17220
************************************************************************ RSC17230
************************************************************************ RSC17240
**                                                                    ** RSC17250
**   CALLED BY.   ERRCK.                                              ** RSC17260
**   ROUTINES CALLED.   SCHDULR, TCLA.                                ** RSC17270
**   CALLING CONVENTIONS.                                             ** RSC17280
**      1. TCBR MUST POINT TO TCB THAT HAS JUST RECEIVED AN INPUT     ** RSC17290
**         MESSAGE AND HAS NOT BEEN PUT IN FORMAT EXPECTED BY A       ** RSC17300
**         MESSAGE PROCESSING PROGRAM.                                ** RSC17310
**   FUNCTIONS.                                                       ** RSC17320
**      1. SHIFT OUT STX AND TERMINAL ADDRESS BYTES IN I/O AREA.      ** RSC17330
**      2. FIND ETX CHARACTER IN I/O AREA, CALCULATE LENGTH OF        ** RSC17340
**         MESSAGE, AND STORE IN SCPBF.                               ** RSC17350
**      3. BLANK OUT ETX AND LRC'S AT END OF MESSAGE.                 ** RSC17360
**      4. TRANSLATE INPUT MESSAGE FROM ASCII-2 TO EBCDIC.            ** RSC17370
**      5. CHECK TO SEE IF MESSAGE IS A TERMINAL COMMAND AND GO TO    ** RSC17380
**         TCLA IF SO.                                                ** RSC17390
**                                                                    ** RSC17400
************************************************************************ RSC17410
         SPACE 5                                                        RSC17420
INFRMT   L     MSGARR,TSTAT                                             RSC17430
         LA    WRKR1,4(MSGARR)                                          RSC17440
         MVC   0(256,WRKR1),2(WRKR1)     SHIFT OUT STX & TML ADDR       RSC17450
         MVC   256(229,WRKR1),258(WRKR1)                                RSC17460
         L     WRKR2,TRTSTBLA            SCAN FOR ETX                   RSC17470
         TRT   0(256,WRKR1),0(WRKR2)     WRKR1 & WRKR2 CAN'T BE R1 & R2 RSC17480
         BC    6,BTS0301                 BRANCH IF FOUND                RSC17490
         TRT   256(226,WRKR1),0(WRKR2)                                  RSC17500
         BC    8,BTS0302                 BRANCH IF NO ETX FOUND         RSC17510
BTS0301  SR    1,WRKR1                   REG 1 CONTAINS MSG LNGTH       RSC17520
         ST    1,0(MSGARR)               STORE LNGTH                    RSC17530
         LR    MSGARR,WRKR1                                             RSC17540
         LA    WRKR1,0(1,MSGARR)                                        RSC17550
         MVC   0(4,WRKR1),=C'    '                                      RSC17560
         L     TABLR,INTBLA                                             RSC17570
         LR    LGTHR,1                                                  RSC17580
         BAL   RETR,TRNSLT              TRANSLATE INCOMING MSG          RSC17590
         CLI   0(MSGARR),C'/'                                           RSC17600
         BE    TCLA                     A '/' INDICATES TERMINAL COMMND RSC17610
         B     SCHDULR                                                  RSC17620
BTS0302  LA    1,480(WRKR1)             COUNT WILL BE 480               RSC17630
         B     BTS0301                                                  RSC17640
         SPACE 5                                                        RSC17650
TRTSTBLA DC    V(TRTSTBL)               ADDR OF TRANSLATE & TEST TABLE  RSC17660
INTBLA   DC    V(IECTRSCI)              ADDR OF TRNSLT TABLE FOR INPUT  RSC17670
         EJECT                                                          RSC17680
*********************************************************************** RSC17690
*********************************************************************** RSC17700
**                                                                   ** RSC17710
**                           M S G P R O C                           ** RSC17720
**                                                                   ** RSC17730
*********************************************************************** RSC17740
*********************************************************************** RSC17750
**                                                                   ** RSC17760
**   CALLED BY.   SCHDULR.                                           ** RSC17770
**   ROUTINES CALLED.   TSKALOC, TERM PROGRAMS(EXTERNAL), SCHDULR.   ** RSC17780
**   CALLING CONVENTIONS.                                            ** RSC17790
**      1. TCBR MUST POINT TO TCB WHICH IS READY TO HAVE A MES-      ** RSC17800
**         SAGE PROCESSING ROUTINE CALLED.                           ** RSC17810
**   FUNCTIONS.                                                      ** RSC17820
**      1. ZERO ECB IN TCB.                                          ** RSC17830
**      2. OBTAIN ADDRESS OF TERM SUBROUTINE ASSOCIATED WITH THIS    ** RSC17840
**         TCB.                                                      ** RSC17850
**      3. LOAD REGISTER 1 WITH ADDRESS OF THIS TCB.                 ** RSC17860
**      4. CALL APPROPRIATE SUBROUTINE.                              ** RSC17870
**                                                                   ** RSC17880
*********************************************************************** RSC17890
         SPACE 5                                                        RSC17900
MSGPROC  B     TSKALOC                  CK IN WITH TASK ALLOCATION      RSC17910
RTNFTSK  MVI   0(TCBR),X'00'            ZERO ECB                        RSC17920
         OI    TSTAT,X'08'              TURN ON MSG PROC BIT            RSC17930
         SR    WRKR1,WRKR1              ZERO WORK REG 1                 RSC17940
         IC    WRKR1,TNO                PUT TERM NBR IN REG 1           RSC17950
         SLL   WRKR1,2(0)               SHIFT LEFF BY 2 SAME AS X=X*4   RSC17960
         L     15,TRMPRGMS(WRKR1)       GET ADDR OF TERM PROG LIST      RSC17970
         LA    1,TPECB(WRKR1)           LOAD PASS REG WITH ECB ADDR     RSC17980
         BALR  14,15                    GO TO THE TERM PROGRAM          RSC17990
         B     SCHDULR                                                  RSC18000
TRMPRGMS DS    0F                       LIST OF TERM PROGRAMS           RSC18010
         DC    V(TERM1)                 ADD OF TERM PROG ONE            RSC18020
         DC    V(TERM2)                 ADD OF TERM PROG TWO            RSC18030
TPECB    DS    0F                                                       RSC18040
         DC    V(TCB1A)                                                 RSC18050
         DC    V(TCB1B)                                                 RSC18060
         EJECT                                                          RSC18070
*********************************************************************** RSC18080
*********************************************************************** RSC18090
**                                                                   ** RSC18100
**                         O U T F R M T                             ** RSC18110
**                                                                   ** RSC18120
*********************************************************************** RSC18130
*********************************************************************** RSC18140
**                                                                   ** RSC18150
**  CALLED BY.   EWRITEFB, WRINIT.                                   ** RSC18160
**  ROUTINES CALLED.   WRITE.                                        ** RSC18170
**  CALLING CONVENTIONS.                                             ** RSC18180
**     1. MSGARR MUST POINT TO SCPBF+4 OF TCB THAT IS GETTING READY  ** RSC18190
**        TO WRITE A MESSAGE.                                        ** RSC18200
**     2. LGTHR MUST CONTAIN THE LENGTH OF THE MESSAGE TO BE TRANS-  ** RSC18210
**        LATED.                                                     ** RSC18220
**  FUNCTIONS.                                                       ** RSC18230
**     1. TRANSLATE OUTGOING MESSAGE FROM EBCDIC TO ASCII-2.         ** RSC18240
**     2. INSERT ETX FOLLOWING LAST BYTE OF MESSAGE.                 ** RSC18250
**     3. BUMP LGTHR BY 1 TO INCLUDE ETX IN THE LENGTH COUNT.        ** RSC18260
**     4. BRANCH TO WRITE ROUTINE.                                   ** RSC18270
**                                                                   ** RSC18280
*********************************************************************** RSC18290
```

```
         SPACE 5                                                        RSC18300
OUTFRMT  L     TABLR,OUTBLA                                             RSC18310
         BAL   RETR,TRNSLT              TRANSLATE OUTGOING MSG          RSC18320
         LA    WRKR1,0(LGTHR,MSGARR)    ADDR OF 1ST BYTE FOLLOWING MSG  RSC18330
         MVI   0(WRKR1),X'03'           INSERT ETX                      RSC18340
         LA    LGTHR,1(LGTHR)           ADD 1 TO MESSAGE LENGTH         RSC18350
         BR    WRTREG                   GO WRITE                        RSC18360
         SPACE 3                                                        RSC18370
TRNSLT   ST    4,SVBASE2                                                RSC18380
         L     DCBR,DCBADR                                              RSC18390
         TRNSLATE (DCBR),(TABLR),(MSGARR),(LGTHR)                       RSC18400
         L     4,SVBASE2                                                RSC18410
         BR    RETR                                                     RSC18420
         SPACE 5                                                        RSC18430
OUTBLA   DC    V(IECTSSCI)              ADDR OF TRNSLT TABLE FOR OUTPUT RSC18440
         EJECT                                                          RSC18450
*********************************************************************** RSC18460
*********************************************************************** RSC18470
**                                                                   ** RSC18480
**                           P R W T A N S                           ** RSC18490
**                                                                   ** RSC18500
*********************************************************************** RSC18510
*********************************************************************** RSC18520
**                                                                   ** RSC18530
**  CALLED BY.   ADDL, ADDT, DISPLAY, DROPL, DROPT.                  ** RSC18540
**  ROUTINES CALLED.   SCHDULR.                                      ** RSC18550
**  CALLING CONVENTIONS.                                             ** RSC18560
**     1. MSGARR MUST POINT TO SCPBF+4 OF TCB TO GET REPLY.          ** RSC18570
**     2. TCBR MUST POINT TO TCB OF TERMINAL GETTING REPLY.          ** RSC18580
**  FUNCTIONS.                                                       ** RSC18590
**     1. SET UP REPLY MESSAGE FROM A TERMINAL COMMAND.              ** RSC18600
**     2. TRANSLATE MESSAGE INTO ASCII-2.                            ** RSC18610
**     3. INSERT ETX AT END OF MESSAGE.                              ** RSC18620
**     4. STORE ADDRESS OF WTANS IN NXTRN1.                          ** RSC18630
**                                                                   ** RSC18640
*********************************************************************** RSC18650
```

```
         SPACE 5                                                        RSC18660
* PREPARE TO WRITE ANSWER ROUTINE...IT HAS 6 POSSIBLE ENTRY POINTS      RSC18670
* PRWTANS1...MEANS TERMINAL BEING DROPPED                               RSC18680
* PRWTANS2...MEANS LINE BEING DROPPED                                   RSC18690
* PRWTANS3...MEANS USER SUPPLIED INCORRECT TERMINAL OR LINE ADD/DROP    RSC18700
* PRWTANS4...MEANS TERMINAL BEING ADDED                                 RSC18710
* PRWTANS5...MEANS LINE BEING ADDED                                     RSC18720
         SPACE 5                                                        RSC18730
PRWTANS1 MVC   0(12,MSGARR),=C'TERMINAL    '  STORE WORD IN MSG AREA    RSC18740
         MVC   9(2,MSGARR),NTSTA                                        RSC18750
         B     STM00050                                                 RSC18760
         SPACE 5                                                        RSC18770
PRWTANS2 MVC   0(12,MSGARR),=C'  LINE      '  STORE WORD IN MSG AREA    RSC18780
         MVC   9(2,MSGARR),NTSTA                                        RSC18790
STM00050 MVC   12(7,MSGARR),=C'DROPPED'      STORE WORD IN MSG AREA     RSC18800
         MVI   19(MSGARR),X'03'         STORE ETX IN MSG AREA           RSC18810
         LA    LGTHR,19(0)                                              RSC18820
         SPACE 5                                                        RSC18830
PRWTANS3 LA    WRKR1,WTANS              PUT ADDR OF WTANS IN NXTRN1     RSC18840
         ST    WRKR1,NXTRN1                                             RSC18850
         OI    TSTAT,X'02'              TURN ON IOS1                    RSC18860
         L     TABLR,OUTBLA             LOAD TRNS TBL REG               RSC18870
         BAL   RETR,TRNSLT              GO TRANSLATE                    RSC18880
         B     SCHDULR                  WAIT FOR TURN                   RSC18890
         SPACE 5                                                        RSC18900
PRWTANS4 MVC   0(12,MSGARR),=C'TERMINAL    '  STORE WORD IN MSG AREA    RSC18910
         MVC   9(2,MSGARR),NTSTA        STORE TRM NO. IN MSG AREA       RSC18920
         B     STM00051                                                 RSC18930
         SPACE 5                                                        RSC18940
PRWTANS5 MVC   0(12,MSGARR),=C'     LINE   '  STORE WORD IN MSG AREA    RSC18950
         MVC   9(2,MSGARR),NTSTA        STORE LINE NO. IN MSG AREA      RSC18960
STM00051 MVC   12(5,MSGARR),=C'ADDED'   STORE WORD IN MSG AREA          RSC18970
         MVI   17(MSGARR),X'03'         STORE ETX IN MSG AREA           RSC18980
         LA    LGTHR,17(0)              LOAD LENGTH REG                 RSC18990
         B     PRWTANS3                                                 RSC19000
         EJECT                                                          RSC19010
```

```
*********************************************************************** RSC19020
*********************************************************************** RSC19030
**                                                                   ** RSC19040
**                          R E A D F B                              ** RSC19050
**                                                                   ** RSC19060
*********************************************************************** RSC19070
*********************************************************************** RSC19080
**                                                                   ** RSC19090
**  CALLED BY.   SCHDULR.                                            ** RSC19100
**  ROUTINES CALLED.   SCHDULR, SIOCK.                               ** RSC19110
**  CALLING CONVENTIONS.                                             ** RSC19120
**     1. TCBR MUST POINT TO TCB THAT WILL READ.                     ** RSC19130
**  FUNCTIONS.                                                       ** RSC19140
**     1. LOAD MSGARR WITH ADDRESS OF I/O AREA, LOAD LGTHR WITH 490  ** RSC19150
**        (LENGTH OF READ), & LOAD TRMLR WITH ADDRESS OF TERMINAL    ** RSC19160
**        LIST FOR READ MACRO INSTRUCTION.                           ** RSC19170
**     2. PUT ADDRESS OF ERRCK IN NXTRN1 AND ADDRESS OF MSGPROC IN   ** RSC19180
**        NXTRN2.                                                    ** RSC19190
**     3. ZERO DECB.                                                 ** RSC19200
**     4. ISSUE READ MACRO.                                          ** RSC19210
**     5. RETURN TO SCHEDULER.                                       ** RSC19220
**                                                                   ** RSC19230
*********************************************************************** RSC19240
         SPACE 5                                                        RSC19250
READFB   L     DECBR,ADECB              LOAD ADDR OF DECB               RSC19260
         L     MSGARR,TSTAT             LOAD ADDR OF MSG AREA           RSC19270
         LA    MSGARR,4(MSGARR)         ALLOW FOR COUNT                 RSC19280
         LA    LGTHR,490(0)              LOAD LNGTH FOR READ FB         RSC19290
         L     TRMLR,TLST               LOAD ADDR OF TERMINAL LIST      RSC19300
         LA    WRKR1,ERRCK              PUT ADDR OF ERRCK IN NXTRN1     RSC19310
         ST    WRKR1,NXTRN1                                             RSC19320
         LA    WRKR1,MSGPROC            PUT ADDR OF MSGPROC IN NXTRN2   RSC19330
         ST    WRKR1,NXTRN2             ZERO  BIOS2 ALSO                RSC19340
         MVC   DECSDECB,=F'0'           ZERO ECB IN DECB                RSC19350
         ST    TCBR,TADECB              PUT ADDR OF TCB IN DECB         RSC19360
* ISSUE READ FB                                                         RSC19370
```

```
         READ  (DECBR),TBR,,(MSGARR),(LGTHR),(TRMLR),,MF=E              RSC19380
         BAL   RETREG,SIOCK            CCW CHECK ROUTINE                RSC19390
         B     SCHDULR                 BRANCH                           RSC19400
         EJECT                                                          RSC19410
*********************************************************************** RSC19420
*********************************************************************** RSC19430
**                                                                   ** RSC19440
**                           R E S C H D L R                         ** RSC19450
**                                                                   ** RSC19460
*********************************************************************** RSC19470
*********************************************************************** RSC19480
**                                                                   ** RSC19490
**  CALLED BY.  SPCWRT, WTANS (BOTH VIA SCHDULR).                    ** RSC19500
**  ROUTINES CALLED.   SPCPOLL.                                      ** RSC19510
**  CALLING CONVENTIONS.                                             ** RSC19520
**     1. TCBR MUST POINT TO TCB THAT IS BEING RESCHEDULED FOR       ** RSC19530
**        EITHER A READ FULL BUFFER OR A READ MANUAL INPUT.          ** RSC19540
**  FUNCTIONS.                                                       ** RSC19550
**     1. RESCHEDULES A TERMINAL FOR THE PARTICULAR TYPE OF READ     ** RSC19560
**        COMMAND IT WAS EXECUTING WHEN IT ENTERED THE TERMINAL      ** RSC19570
**        COMMAND.                                                   ** RSC19580
**                                                                   ** RSC19590
*********************************************************************** RSC19600
         SPACE 5                                                        RSC19610
RESCHDLR TM    TPCW+3,X'04'            WAS IT READ MI                   RSC19620
         BO    STM00001                YES.BRANCH                       RSC19630
         LA    WRKR1,READFB            NO.PUT ADDR OF SPCPOLL IN NXTRN1 RSC19640
         ST    WRKR1,NXTRN2                                             RSC19650
         OI    BIOS2,X'02'                                              RSC19660
         B     SPCPOLL                                                  RSC19670
STM00001 LA    WRKR1,MSGPROC                                            RSC19680
         ST    WRKR1,NXTRN2                                             RSC19690
         B     SPCPOLL                                                  RSC19700
         EJECT                                                          RSC19710
*********************************************************************** RSC19720
*********************************************************************** RSC19730
```

```
**                                                                    ** RSC19740
**                          S C H D U L R                             ** RSC19750
**                                                                    ** RSC19760
************************************************************************ RSC19770
************************************************************************ RSC19780
**                                                                    ** RSC19790
**  CALLED BY.   INIT, DECBWT, ERRCK, INFRMT, MSGPROC, PRWTANS,       ** RSC19800
**       READFB, SPCPOLL, STOP, TCLA, TECBWT, WRITE.                  ** RSC19810
**  ROUTINES CALLED.   EWRITEFB, MSGPROC, READFB, SPCPOLL, SPCWRT,    ** RSC19820
**       TECBWT, WRINIT, WTANS.                                       ** RSC19830
**  ABENDS.   1.                                                      ** RSC19835
**  CALLING CONVENTIONS.                                              ** RSC19840
**     1. NONE.                                                       ** RSC19850
**  FUNCTIONS.                                                        ** RSC19860
**     1. TO SCHEDULE AND SERVICE EACH TERMINAL FOR BOTH 2260 I/O     ** RSC19870
**        AND MESSAGE PROCESSING.                                     ** RSC19880
**     2. TO BYPASS SERVICING TCB'S WHICH ARE DROPPED, MESSAGE PRO-   ** RSC19890
**        CESSING OR DOING I/O.                                       ** RSC19900
**                                                                    ** RSC19910
**                                                                    ** RSC19920
**  USE OF CONTROL BLOCKS.                                            ** RSC19930
**                                                                    ** RSC19940
**     THE SCHEDULER OBTAINS THE NEXT LINE CONTROL BLOCK (LCB) TO     ** RSC19950
**  BE SERVICED FROM THE SCHEDULER CONTROL BLOCK (SCB), AND FROM      ** RSC19960
**  THE LCB, THE NEXT TERMINAL CONTROL BLOCK (TCB) IS OBTAINED.       ** RSC19970
**  THE TCB CONTAINS ALL STATUS CONCERNING THE PARTICULAR TERMINAL,   ** RSC19980
**  SUCH AS DROPPED OR NOT DROPPED, TERMINAL NUMBER, ADDRESS OF       ** RSC19990
**  TERMINAL LIST, ERROR INFORMATION FROM THE LAST I/O OPERATION,     ** RSC20000
**  THE ADDRESS OF THE ROUTINE TO BE USED NEXT AND OTHER INFORMATION  ** RSC20010
**  CONCERNING THIS TERMINAL.                                         ** RSC20020
**     THE LCB CONTAINS A POINTER TO THE ADDRESS OF THE TCB WHICH     ** RSC20030
**  IS TO BE SERVICED NEXT, THE ADDRESS OF THE DECB FOR THIS LINE,    ** RSC20040
**  THE LINE DROPPED OR NOT DROPPED STATUS AND THE ADDRESS OF EACH    ** RSC20050
**  TCB ON THIS LINE.                                                 ** RSC20060
**     THE SCB CONTAINS A POINTER TO THE ADDRESS OF THE LCB TO BE     ** RSC20070
**  SERVICED NEXT AND THE ADDRESS OF EACH LCB.                        ** RSC20080
**                                                                    ** RSC20090
**                                                                    ** RSC20100
**  EXPLANATION OF THE SCHEDULING ALGORITHM.                          ** RSC20110
**                                                                    ** RSC20120
**     THE SCHEDULER WILL GENERALLY DO THE FOLLOWING THINGS..         ** RSC20130
**        1. LOOK AT THE NEXT TCB ON A LINE TO SEE IF IT NEEDS AT-    ** RSC20140
**        TENTION.                                                    ** RSC20150
**        2. IF SO, BRANCH TO THAT SERVICE ROUTINE.                   ** RSC20160
**        3. IF NOT, GET THE NEXT TCB ON THE NEXT LINE.               ** RSC20170
**     AFTER A TERMINAL HAS BEEN LOOKED AT FOR SERVICE REQUIRE-       ** RSC20180
**  MENTS BY THE SCHEDULER, A TCB ON THE NEXT LINE WILL BE NEXT IN    ** RSC20190
**  LINE FOR CONSIDERATION BY THE SCHEDULER. WITHIN EACH LCB, THE     ** RSC20200
**  TCB'S FOR THAT LINE ARE SEQUENTIALLY LOOKED AT IN TURN. IF        ** RSC20210
**  EACH LINE IS EITHER                                               ** RSC20220
**     (1) BUSY OR                                                    ** RSC20230
**     (2) EVERY TCB ON THE LINE IS MESSAGE PROCESSING OR DROPPED     ** RSC20240
**     AND AT LEAST ONE LINE IS BUSY,                                 ** RSC20250
**  THEN THE SCHEDULER WILL GO TO THE DECBWT ROUTINE TO WAIT FOR      ** RSC20260
**  THE COMPLETION OF AT LEAST ONE I/O OPERATION.                     ** RSC20270
**     IF EACH TERMINAL ON EACH LINE IS EITHER MESSAGE PROCESSING     ** RSC20280
**  OR DROPPED THE SCHEDULER WILL GO TO THE TECBWT ROUTINE TO WAIT    ** RSC20290
**  FOR THE COMPLETEION OF AT LEAST ONE MESSAGE PROCESSING PROGRAM.   ** RSC20300
**                                                                    ** RSC20310
************************************************************************ RSC20320
                                                                         RSC20330
         SPACE 5                                                         RSC20340
*                              *****************                         RSC20350
*                              *    I - 1      *                         RSC20360
*                              *****************                         RSC20370
         SPACE 3                                                         RSC20380
SCHDULR  L     LCBR,NXTLCB               ADDR OF PTR TO NEXT LCB         RSC20390
         L     LCBR,0(LCBR)              ADDR OF NEXT LCB                RSC20400
         TM    LNSTAT,X'01'              IF LINE NOT DOWN                RSC20410
         BZ    BTS0001                   BRANCH                          RSC20420
BTS00005 LA    RETR,SCHDULR              BYPASS THIS LINE                RSC20430
         B     UPDTLCB                                                   RSC20440
BTS0001  L     TCBR,NXTTCB               ADDR OF PTR TO NEXT TCB
```

```
         L     TCBR,0(TCBR)           ADDR OF NEXT TCB                 RSC20450
         TM    TSTAT,X'01'            IS TERM DROPPED                  RSC20460
         BZ    BTS0002                BRANCH NO                        RSC20470
UTCBLCBS BAL   RETR,UPDTTCB                                            RSC20480
         B     BTS00005                                                RSC20490
         SPACE 3                                                       RSC20500
*                       *****************                              RSC20510
*                       *    I - 7      *                              RSC20520
*                       *****************                              RSC20530
         SPACE 3                                                       RSC20540
BTS0002  TM    TSTAT,X'02'            IOS1 ON                          RSC20550
         BZ    PRCESS                 BRANCH NO                        RSC20560
         L     DECBR,ADECB            ADDR OF DECB FOR THIS LINE       RSC20570
         TM    DECSDECB,X'40'         IF LINE NOT BUSY                 RSC20580
         BO    LNOTBSY                BRANCH                           RSC20590
         SPACE 3                                                       RSC20600
*                       *****************                              RSC20610
*                       *    I - 9      *                              RSC20620
*                       *****************                              RSC20630
         SPACE 3                                                       RSC20640
         L     SAVR1,NXTLCB           SAVE ADDR OF OLD LCB             RSC20650
         L     SAVR2,NXTTCB                                            RSC20660
BTS0003  BAL   RETR,UPDTLCB           GET NEXT LCB                     RSC20670
         L     LCBR,NXTLCB                                             RSC20680
         CR    SAVR1,LCBR             ARE WE BACK TO THE ORIGINAL LCB RSC20690
         BE    DECBWT                 WAIT ON 2260 I/O                 RSC20700
STAT     L     LCBR,0(LCBR)           ADDR OF LCB                      RSC20710
         TM    LNSTAT,X'01'           IF LINE IS DROPPED               RSC20720
         BO    BTS0003                GET NEXT LCB                     RSC20730
         L     DECBR,ADECB            GET ADDR OF DECB FOR THIS LINE   RSC20740
         TM    DECSDECB,X'40'         IF NOT COMPLETE                  RSC20750
         BZ    BTS0003                GET NEXT LCB                     RSC20760
         BAL   RETR,UPDTTCB           GET NEXT TCB                     RSC20770
         L     TCBR,NXTTCB                                             RSC20780
         CR    SAVR2,TCBR             ALL TCBS FOR THIS LINE CHECKED   RSC20790
         BE    BTS0003                BRANCH YES                       RSC20800
         L     TCBR,0(TCBR)            ADDR OF TCB                      RSC20810
         TM    TSTAT,X'01'             IF TERM DROPPED                  RSC20820
         BO    STAT                    NEXT TCB                         RSC20830
         TM    TECB,X'40'              IF MSG PROCESSING                RSC20840
         BZ    STAT                    NEXT TCB                         RSC20850
         B     SCHDULR                 ELSE PROCESS CURRENT LCB, TCB    RSC20860
         SPACE 3                                                        RSC20870
*                        *****************                              RSC20880
*                        *    I - 13     *                              RSC20890
*                        *****************                              RSC20900
         SPACE 3                                                        RSC20910
LNOTBSY  LR    SAVR2,TCBR                                               RSC20920
         L     TCBR,TADECB                                              RSC20930
         MVC   TSECB(8),DECSDECB        SAVE I/O ERROR INFO IN TCB      RSC20940
         MVC   TSNS1(4),DECSENSO                                        RSC20950
         MVC   TSTSFLGS(8),DECFLAGS                                     RSC20960
         LR    TCBR,SAVR2                                               RSC20970
BTS0004  L     SAVR1,NXTRN1             GET ADDR OF NXTRN1 FOR THIS TCB RSC20980
         BAL   RETR,UPDTTCB                                             RSC20990
         LR    RETR,SAVR1                                               RSC21000
         B     UPDTLCB                                                  RSC21010
         SPACE 3                                                        RSC21020
*                        *****************                              RSC21030
*                        *    I - 18     *                              RSC21040
*                        *****************                              RSC21050
         SPACE 3                                                        RSC21060
PRCESS   TM    TECB,X'40'               IS MSG PROC COMPLETE            RSC21070
         BO    TMSPF                    IF YES GO TO TEST MSGP FINISHED RSC21080
*        ARE ALL TERMS ON ALL LINES MSG PROCESSING                      RSC21090
         L     SAVR1,NXTLCB             SAVE CRRNT LCB ADDR FOR TESTING RSC21100
         L     SAVR2,NXTTCB             SAVE CRRNT TCB ADDR FOR TESTING RSC21110
GETNXTT  MVC   SNXTTCB,NXTTCB           SAVE FOR BACKING UP             RSC21120
         BAL   RETR,UPDTTCB             GET NEXT TERM                   RSC21130
         L     TCBR,NXTTCB                                              RSC21140
         CR    SAVR2,TCBR               IS CYCLE ON TERMS COMPLETE      RSC21150
         BE    GETNXTL                  IF YES GET NEXT LINE            RSC21160
```

```
TSTTRM   L     TCBR,0(TCBR)            GET THE ADDR OF THE TCB          RSC21170
         TM    TSTAT,X'01'             IS THE TERM DOWN                 RSC21180
         BO    GETNXTT                 IF YES GO GET NEXT TERM          RSC21190
         TM    TECB,X'40'              IS THIS TERM MSG PROC            RSC21200
         BO    GETNXTT                 IF YES GO GET NEXT TERM          RSC21210
GRFSTAT  L     LCBR,NXTLCB             GET READY FOR STAT               RSC21220
         L     SAVR2,SNXTTCB           POINT TCB BACK ON FOR STAT       RSC21230
         ST    SAVR2,NXTTCB                                             RSC21240
         B     STAT                                                     RSC21250
GETNXTL  BAL   RETR,UPDTLCB            GET NEXT LINE                    RSC21260
         L     LCBR,NXTLCB                                              RSC21270
         CR    SAVR1,LCBR              IS CYCLE ON LINES COMPLETE       RSC21280
         BE    TECBWT                  IF YES GO WAIT                   RSC21290
         L     LCBR,0(LCBR)            GET ADDR OF LCB                  RSC21300
         TM    LNSTAT,X'01'            IS LINE DOWN                     RSC21310
         BE    GETNXTL                 IF YES GO GET NEXT LINE          RSC21320
         L     TCBR,NXTTCB             IF NO GET NEXT TERM              RSC21330
         LR    SAVR2,TCBR              SAVE CRRNT TERM                  RSC21340
*        DOWN DATE THE TCB POINTER FOR STAT                             RSC21350
DWDTTCB  LA    WRKR1,ATCB1             IS THIS TERM FIRST TERM IN LIST  RSC21360
         CR    TCBR,WRKR1                                               RSC21370
         BE    GTLTCB                  IF YES GO GET LAST TERM  IN LST  RSC21380
         S     TCBR,=F'4'              IF NOT GET PREVIOUS TERM         RSC21390
SNTCB    ST    TCBR,SNXTTCB            SAVE THIS TERM                   RSC21400
         LR    TCBR,SAVR2                                               RSC21410
         B     TSTTRM                  GO TEST TERM                     RSC21420
GTLTCB   TM    0(TCBR),X'80'           IS FST TERM ALSO LAST TERM       RSC21430
         BE    SNTCB                   IF IT IS ONLY TERM GO TO SNLCB   RSC21440
GETLSTCB LA    TCBR,4(TCBR)            GET THE NEXT TERM IN LIST        RSC21450
         TM    0(TCBR),X'80'           IS THIS THE LAST TERM            RSC21460
         BE    SNTCB                   IF YES GO SAVE THE TERM          RSC21470
         B     GETLSTCB                IF NOT GO GET NEXT TERM          RSC21480
         SPACE 3                                                        RSC21490
*                      *****************                                RSC21500
*                      *    I - 22     *                                RSC21510
*                      *****************                                RSC21520
```

```
         SPACE 3                                                        RSC21530
TMSPF    TM    TSTAT,X'08'             JUST FINISHED MSG PROC           RSC21540
         BZ    BTS0004                  IF NO GET READY FOR NXT RTN     RSC21550
         SPACE 3                                                        RSC21560
*                        *****************                              RSC21570
*                        *     I - 27    *                              RSC21580
*                        *****************                              RSC21590
         SPACE 3                                                        RSC21600
AWTDNXT  NI    TSTAT,X'F7'              TURN OFF BIT FOUR OF TSTAT      RSC21610
         TM    TPCW+3,X'10'            IS THE USER ASKING FOR PROCESING RSC21620
         BZ    USERWIO                 IF NO USE WANTS IO               RSC21630
         BAL   RETR,UPDTTCB            GO UPDATE TCB                    RSC21640
         LA    RETR,MSGPROC            UPDATE LCB THEN                  RSC21650
         B     UPDTLCB                 GO TO MSG PROC RTN               RSC21660
USERWIO  OI    TSTAT,X'02'             TURN ON IO STATUS BIT (IOS1)     RSC21670
         TM    TPCW+3,X'08'            WAS THE IO REQUEST A EWFB        RSC21680
         BO    IOREWFB                 IF YES GO TO IO REQUEST WAS EWFB RSC21690
         TM    TPCW+3,X'04'            WAS IT A RMI REQUEST             RSC21700
         BO    IORRMI                  IF YES GO SET UP RMI             RSC21710
         TM    TPCW+3,X'02'            WAS IT A RFB REQUEST             RSC21720
         BO    IORRFB                  IF YES GO SET UP RFB             RSC21730
         TM    TPCW+3,X'01'            WAS IT A STOP REQUEST            RSC21740
         BO    IORSTOP                 IF YES GO SET UP STOP            RSC21750
         LA    1,1(0)                                                   RSC21760
         B     ABEND                   IF NO SOMETHING IS BAD WRONG     RSC21770
         SPACE 3                                                        RSC21780
*                        *****************                              RSC21790
*                        *     I - 32    *                              RSC21800
*                        *****************                              RSC21810
         SPACE 3                                                        RSC21820
IOREWFB  TM    TSTAT,X'04'              WAS LAST I/O OPER A WRITE       RSC21830
         BO    LOWAIOP                 IF YES GO TO LST OP WAS A IO OP  RSC21840
         LA    WRKR1,EWRITEFB          PUT EWRFB IN NXT RTN ONE         RSC21850
         ST    WRKR1,NXTRN1                                             RSC21860
         OI    TSTAT,X'04'             TURN ON LAST OPER WAS A IO OP    RSC21870
         B     UTCBLCBS                UPDATE TCB LCB THEN GOTO SCHDLR  RSC21880
```

```
LOWAIOP  LA    WRKR1,SPCPOLL           PUT ADDR OF SPEC POLL IN NXTRN1  RSC21890
         ST    WRKR1,NXTRN1                                             RSC21900
         LA    WRKR1,EWRITEFB          PUT ADDR OF EWFB IN NXTRN2       RSC21910
         ST    WRKR1,NXTRN2                                             RSC21920
         L     WRKR1,TSTAT              LOAD SCPBF AREA                 RSC21930
         ST    WRKR1,TSPAREA                                            RSC21940
HBTS     OI    BIOS2,X'02'             TURN ON IOS2 BIT                 RSC21950
         B     UTCBLCBS                UPDATE TCB LCB THEN GOTO SCHDLR  RSC21960
         SPACE 3                                                        RSC21970
*                       *****************                               RSC21980
*                       *     I - 38    *                               RSC21990
*                       *****************                               RSC22000
         SPACE 3                                                        RSC22010
IORRMI   LA    WRKR1,SPCPOLL           PUT ADDR OF SPEC POLL IN NXTRN1  RSC22020
         ST    WRKR1,NXTRN1                                             RSC22030
         LA    WRKR1,MSGPROC           PUT ADDR OF MSG PROC IN NXTRN2   RSC22040
         ST    WRKR1,NXTRN2                                             RSC22050
         L     WRKR1,TSTAT              LOAD SCPBF AREA                 RSC22060
         LA    WRKR1,4(WRKR1)           BUMP BY 4                       RSC22070
         ST    WRKR1,TSPAREA                                            RSC22080
         NI    BIOS2,X'FD'             TURN OFF IOS2 BIT                RSC22090
TFIOS2   NI    TSTAT,X'FB'             TURN OFF IOLW LST OP NOT AWRITE  RSC22100
         B     UTCBLCBS                UPDATE TCB LCB THEN GOTO SCHDLR  RSC22110
         SPACE 3                                                        RSC22120
*                       *****************                               RSC22130
*                       *     I - 41    *                               RSC22140
*                       *****************                               RSC22150
         SPACE 3                                                        RSC22160
IORRFB   LA    WRKR1,SPCPOLL           PUT ADDR OF SPEC POLL IN NXTRN1  RSC22170
         ST    WRKR1,NXTRN1                                             RSC22180
         LA    WRKR1,READFB            PUT ADDR OF RFB IN NXTRN2        RSC22190
         ST    WRKR1,NXTRN2                                             RSC22200
         L     WRKR1,TSTAT              LOAD SCPBF AREA                 RSC22210
         LA    WRKR1,4(WRKR1)                                           RSC22220
         ST    WRKR1,TSPAREA                                            RSC22230
         OI    BIOS2,X'02'             TURN ON IOS2 BIT                 RSC22240
         B     TFIOS2                   GO TURN OFF IOS2 BIT           RSC22250
         SPACE 3                                                        RSC22260
*                    *****************                                  RSC22270
*                    *    I - 45     *                                  RSC22280
*                    *****************                                  RSC22290
         SPACE 3                                                        RSC22300
IORSTOP  TM    TSTAT,X'04'               WAS LAST I/O OPER A WRITE      RSC22310
         BO    SUNR                     IF YES GO SET UP NEXT RTNS     RSC22320
         LA    WRKR1,WRINIT              PUT ADDR OF SPC POLL IN NXTRN1 RSC22330
         ST    WRKR1,NXTRN1                                             RSC22340
         B     HBTS                     HEAD BACK TO THE SCHDULAR      RSC22350
SUNR     LA    WRKR1,SPCPOLL            PUT ADDR OF SPC POL IN NXT RTN1 RSC22360
         ST    WRKR1,NXTRN1                                             RSC22370
         LA    WRKR1,WRINIT             PUT ADDR OF WRITE INT IN NXTRN1 RSC22380
         ST    WRKR1,NXTRN2                                             RSC22390
         L     WRKR1,TSTAT               LOAD SCPBF AREA                RSC22400
         ST    WRKR1,TSPAREA                                            RSC22410
         B     HBTS                     HEAD BACK TO SCHDULAR          RSC22420
         SPACE 3                                                        RSC22430
*                                                                       RSC22440
*        UPDATE LCB ROUTINE                                             RSC22450
*                                                                       RSC22460
         SPACE 3                                                        RSC22470
UPDTLCB  ST    LCBR,SAVLCBR                                             RSC22480
         L     LCBR,NXTLCB                                              RSC22490
         TM    0(LCBR),X'80'             LAST ENTRY                     RSC22500
         BO    FRSTLCB                   BRANCH YES                     RSC22510
         LA    LCBR,4(LCBR)              UPDATE ADDR OF PTR BY 4        RSC22520
STNXTLCB ST    LCBR,NXTLCB                                              RSC22530
         L     LCBR,SAVLCBR                                             RSC22540
         BR    RETR                                                     RSC22550
FRSTLCB  LA    LCBR,ALCB1                ADDR OF PTR TO 1ST LCB         RSC22560
         B     STNXTLCB                                                 RSC22570
SAVLCBR  DC    F'0'                                                     RSC22580
         SPACE 3                                                        RSC22590
*                                                                       RSC22600
*        UPDATE TCB ROUTINE                                             RSC22610
*                                                                       RSC22620
         SPACE 3                                                        RSC22630
UPDTTCB  ST    TCBR,SAVTCBR                                             RSC22640
         L     TCBR,NXTTCB                                              RSC22650
         TM    0(TCBR),X'80'            IS THIS THE LAST ENTRY          RSC22660
         BO    FRSTTCB                  IF YES GO RESET POINTER         RSC22670
         LA    TCBR,4(TCBR)             IF NO SET POINTER TO NEXT ENTRY RSC22680
STNXTTCB ST    TCBR,NXTTCB                                              RSC22690
         L     TCBR,SAVTCBR                                             RSC22700
         BR    RETR                                                     RSC22710
FRSTTCB  LA    TCBR,ATCB1               RESET POINTER TO FIRST ENTRY    RSC22720
         B     STNXTTCB                                                 RSC22730
SAVTCBR  DC    F'0'                                                     RSC22740
         SPACE 3                                                        RSC22750
*                                                                       RSC22760
*        ABEND RTN                                                      RSC22770
*                                                                       RSC22780
         SPACE 3                                                        RSC22790
ABEND    ABEND (1),DUMP                                                 RSC22800
*                                                                       RSC22810
         EJECT                                                          RSC22820
*********************************************************************** RSC22830
*********************************************************************** RSC22840
**                                                                   ** RSC22850
**                          S I O C K                                ** RSC22860
**                                                                   ** RSC22870
*********************************************************************** RSC22880
*********************************************************************** RSC22890
**                                                                   ** RSC22900
**  CALLED BY.  READFB, SPCPOLL, WRITE.                              ** RSC22910
**  ROUTINES CALLED.   RETURNS TO THE ABOVE ROUTINES.                ** RSC22920
**  ABENDS.   2.                                                     ** RSC22930
**  CALLING CONVENTIONS.                                             ** RSC22940
**     1. RETR MUST HAVE RETURN ADDRESS.                             ** RSC22950
**     2. REGISTER 15 MUST CONTAIN THE RETURN CODE PLACED IN IT BY   ** RSC22960
**        THE BTAM MODULE USED TO START AN I/O OPERATION.           ** RSC22970
**  FUNCTIONS.                                                       ** RSC22980
**     1. AFTER A READ OR WRITE INSTRUCTION, REGISTER 15 IS CHECKED  ** RSC22990
**        TO SEE IF IT IS NON-ZERO. IF SO, THE PROGRAM ABENDS BE-    ** RSC23000
**        CAUSE IT IS AN ABNORMAL RETURN CODE. IF NOT, NORMAL        ** RSC23010
**        RETURN TO THE CALLING ROUTINE IS PERFORMED.                ** RSC23020
**                                                                   ** RSC23030
*********************************************************************** RSC23040
                                                                        RSC23050
         SPACE 5                                                        RSC23060
SIOCK    LTR   15,15                  CHECK CCW JUST BUILT              RSC23070
         BNZ   SIO00001                                                 RSC23080
         BR    RETR                   RETURN TO CALLING AREA            RSC23090
SIO00001 LA    1,2(0)                                                   RSC23100
         B     ABEND                  BRANCH                            RSC23110
         EJECT                                                          RSC23120
*********************************************************************** RSC23130
*********************************************************************** RSC23140
**                                                                   ** RSC23150
**                          S P C P O L L                            ** RSC23160
**                                                                   ** RSC23170
*********************************************************************** RSC23180
*********************************************************************** RSC23190
**                                                                   ** RSC23200
**  CALLED BY.   RESCHDLR, SCHDULR.                                  ** RSC23210
**  ROUTINES CALLED.   SCHDULR, SIOCK.                               ** RSC23220
**  CALLING CONVENTIONS.                                             ** RSC23230
**     1. TCBR MUST POINT TO TCB FOR WHICH THE SPECIFIC POLL WILL    ** RSC23240
**        BE PERFORMED.                                              ** RSC23250
**  FUNCTIONS.                                                       ** RSC23260
**     1. LOAD ALL REGISTERS (DECBR,MSGARR, & TRMLR) NECESSARY TO    ** RSC23270
**        PERFORM THE READ OPERATION.                                ** RSC23280
**     2. STORE ADDRESS OF ERRCK IN NXTRN1.                          ** RSC23290
**     3. ZERO DECB.                                                 ** RSC23300
**     4. PERFORM READ.                                              ** RSC23310
**     5. RETURN TO SCHDULR.                                         ** RSC23320
**
```

```
*********************************************************************** RSC23330
         SPACE 5                                                        RSC23340
SPCPOLL  L     DECBR,ADECB                  LOAD ADDR OF DECB           RSC23350
         L     MSGARR,TSPAREA                LOAD ADDR OF SPCPOLL BUF   RSC23360
         LA    MSGARR,0(MSGARR)              ZERO FIRST BYTE FOR BTAM   RSC23370
         L     TRMLR,TLST                   LOAD ADDR OF TERMINAL LIST  RSC23380
         LA    WRKR1,ERRCK                  PUT ADDR OF ERRCK IN NXTRN1 RSC23390
         ST    WRKR1,NXTRN1                                             RSC23400
         ST    TCBR,TADECB                  PUT ADDR OF TCB IN DECB     RSC23410
         MVC   DECSDECB,=F'0'               ZERO ECB IN DECB            RSC23420
* ISSUE READ MI                                                         RSC23430
         READ  (DECBR),TIR,,(MSGARR),490,(TRMLR),,MF=E                  RSC23440
         BAL   RETREG,SIOCK                 CCW CHECK ROUTINE           RSC23450
         B     SCHDULR                      BRANCH                      RSC23460
         EJECT                                                          RSC23470
*********************************************************************** RSC23480
*********************************************************************** RSC23490
**                                                                   ** RSC23500
**                            S P C W R T                            ** RSC23510
**                                                                   ** RSC23520
*********************************************************************** RSC23530
*********************************************************************** RSC23540
**                                                                   ** RSC23550
**  CALLED BY.   ERRCK(VIA SCHDULR).                                 ** RSC23560
**  ROUTINES CALLED.   RESCHDLR, WRITE.                              ** RSC23570
**  CALLING CONVENTIONS.                                             ** RSC23580
**     1. TCBR MUST CONTAIN ADDRESS OF TCB WHOSE TERMINAL WAS TO     ** RSC23590
**        PERFORM AN RMI BUT HAD NO START SYMBOL WHEN S/E WAS HIT.   ** RSC23600
**  FUNCTIONS.                                                       ** RSC23610
**     1. LOAD APPROPRIATE REGISTERS (DECBR,MSGARR,LGTHR & TRMLR)    ** RSC23620
**        FOR WRITE.                                                 ** RSC23630
**     2. STORE ADDRESS OF ERRCK IN NXTRN1.                          ** RSC23640
**     3. STORE ADDRESS OF RESCHDLR IN NXTRN2.                       ** RSC23650
**     4. MESSAGE WRITTEN WILL BE 'DO NOT ERASE START SYMBOL'.       ** RSC23660
**                                                                   ** RSC23670
*********************************************************************** RSC23680
```

```
         SPACE 5                                                        RSC23690
SPCWRT   L     DECBR,ADECB             LOAD ADDR OF DECB                RSC23700
         LA    MSGARR,WORDSPEC         LOAD ADDR OF MSG AREA            RSC23710
         LA    LGTHR,SPCLGTH(0)        LOAD LENGTH                      RSC23720
         L     TRMLR,TLST              LOAD ADDR OF TERMINAL LIST       RSC23730
         L     WRKR1,TSTAT              BLANK OUT ETX IN BUFFER SO THAT RSC23740
         MVI   6(WRKR1),X'40'           ERRCK WILL EXECUTE PROPERLY     RSC23750
         LA    WRKR1,ERRCK              PUT ADR OF ERRCK IN NXTRN1      RSC23760
         ST    WRKR1,NXTRN1                                             RSC23770
         LA    WRKR1,RESCHDLR           PUT ADDR OF RESCHDLR IN NXTRN2  RSC23780
         ST    WRKR1,NXTRN2                                             RSC23790
         OI    BIOS2,X'02'                                      7/18/68 RSC23795
         B     WRITE                   BRANCH                           RSC23800
SPCLGTH  EQU   45                                                       RSC23810
* FOLLOWING HEX MSG IS 'DO NOT ERASE START SYMBOL' WITH                 RSC23820
* A START SYMBOL AND ETX AT THE END OF MSG                              RSC23830
WORDSPEC DC    X'4040A4AF40AEAFB440A5B2A1B3A540B3B4A1B2B440B3B9ADA2AFACXRSC23840
               40404040404040404040404040404040BD03'                    RSC23850
         EJECT                                                          RSC23860
*********************************************************************** RSC23870
*********************************************************************** RSC23880
**                                                                   ** RSC23890
**                            S T O P                                ** RSC23900
**                                                                   ** RSC23910
*********************************************************************** RSC23920
*********************************************************************** RSC23930
**                                                                   ** RSC23940
**  CALLED BY.   TCLA.                                               ** RSC23950
**  ROUTINES CALLED.   SCHDULR.                                      ** RSC23960
**  CALLING CONVENTIONS.                                             ** RSC23970
**     1. TCBR MUST CONTAIN ADDRESS OF TCB WHICH HAS ENTERD 'STOP'.  ** RSC23980
**  FUNCTIONS.                                                       ** RSC23990
**     1. STORE ADDRESS OF WRINIT IN NXTRN1 SO THAT TERMINAL WILL    ** RSC24000
**        BE SCHEDULED TO GET THE INITIAL MESSAGE ON HIS SCREEN.     ** RSC24010
**                                                                   ** RSC24020
*********************************************************************** RSC24030
```

```
         SPACE 5                                                        RSC24040
STOP     LA    WRKR1,WRINIT            PUT WRITE INIT IN NXTRN1         RSC24050
         ST    WRKR1,NXTRN1                                             RSC24060
         OI    TSTAT,X'02'                                              RSC24070
         B     SCHDULR                 GO TO SCHDULR                    RSC24080
         EJECT                                                          RSC24090
************************************************************************ RSC24100
************************************************************************ RSC24110
**                                                                    ** RSC24120
**                               T C L A                              ** RSC24130
**                                                                    ** RSC24140
************************************************************************ RSC24150
************************************************************************ RSC24160
**                                                                    ** RSC24170
**  CALLED BY.   INFRMT.                                              ** RSC24180
**  ROUTINES CALLED.   ADDL, ADDT, DISPLAY, DROPL, DROPT, SCHDULR,    ** RSC24190
**     TERMINAT.                                                      ** RSC24200
**  CALLING CONVENTIONS.                                              ** RSC24210
**     1. MSGARR MUST POINT TO SCPBF+4 OF TERMINAL WHICH ENTERD '/'   ** RSC24220
**        AS THE FIRST CHARACTER OF INPUT MESSAGE.                    ** RSC24230
**  FUNCTIONS.                                                        ** RSC24240
**     1. ANALYZE WHICH '/' COMMAND WAS ENTERED & BRANCH TO THE       ** RSC24250
**        APPROPRIATE COMMAND ROUTINE.                                ** RSC24260
**     2. IF IT IS NONE OF THE VALID COMMANDS, TREAT IT AS A REG-     ** RSC24270
**        ULAR INPUT MESSAGE & BRANCH BACK TO SCHDULR.                ** RSC24280
**                                                                    ** RSC24290
************************************************************************ RSC24300
         SPACE 5                                                        RSC24310
TCLA     CLC   1(4,MSGARR),=C'STOP'    IS MSG STOP                      RSC24320
         BE    STOP                    YES                              RSC24330
         CLC   1(5,MSGARR),=C'DROPT'   IS MSG DROP TERMINAL             RSC24340
         BE    DROPT                                                    RSC24350
         CLC   1(5,MSGARR),=C'DROPL'   IS MSG DROP LINE                 RSC24360
         BE    DROPL                                                    RSC24370
         CLC   1(4,MSGARR),=C'ADDT'    IS MSG ADD TERMINAL              RSC24380
         BE    ADDT                    YES                              RSC24390
         CLC   1(4,MSGARR),=C'ADDL'     IS MSG ADD LINE                 RSC24400
         BE    ADDL                                                     RSC24410
         CLC   1(8,MSGARR),=C'TERMINAT'  IS MSG TERMINATE               RSC24420
         BE    TERMINAT               YES                               RSC24430
         CLC   1(7,MSGARR),=C'DISPLAY'                                  RSC24440
         BE    DISPLAY                                                  RSC24450
         B     SCHDULR                                                  RSC24460
         EJECT                                                          RSC24470
*********************************************************************** RSC24480
*********************************************************************** RSC24490
**                                                                   ** RSC24500
**                         T E C B W T                               ** RSC24510
**                                                                   ** RSC24520
*********************************************************************** RSC24530
*********************************************************************** RSC24540
**                                                                   ** RSC24550
**  CALLED BY.   SCHDULR.                                            ** RSC24560
**  ROUTINES CALLED.   SCHDULR.                                      ** RSC24570
**  CALLING CONVENTIONS.                                             ** RSC24580
**     1. AT LEAST ONE OF THE ECB'S IN THE TCB MUST BE COMPLETE OR   ** RSC24590
**        WILL BE POSTED AS COMPLETE. OTHERWISE, THE ROUTINE WILL    ** RSC24600
**        NEVER FALL THROUGH THE TWAIT. (NEVER USED IN MFT).         ** RSC24610
**  FUNCTIONS.                                                       ** RSC24620
**     1. IF ALL TERMINALS ARE EITHER DROPPED OR ARE MESSAGE PRO-    ** RSC24630
**        CESSING, THE SCHDULR WAITS UNTIL AT LEAST ONE TERMINAL     ** RSC24640
**        NEEDS TO BE SCHEDULED FOR SOMETHING ELSE.                  ** RSC24650
**     2. THE TERMINAL THAT FINISHES FIRST, WILL BE SERVICED BY      ** RSC24660
**        SCHDULR AS SOON AS TECBWT RETURNS TO IT.                   ** RSC24670
**                                                                   ** RSC24680
*********************************************************************** RSC24690
         SPACE 5                                                        RSC24700
TECBWT   TWAIT (TCBR),ECBLIST=ELLX2     WAIT FOR ANY EVT TO COMPLETE    RSC24710
         L     LCBR,TLCB                                                RSC24720
         LA    LCBR,0(LCBR)                                             RSC24730
         LA    TCBR,0(TCBR)           ZERO HIGH ORDER BYTE              RSC24740
         LA    WRKR1,ATCB1            PTR TO ADDR OF TCB1               RSC24750
```

```
STM00030 L     WRKR2,0(WRKR1)          ADDR OF TCB1                   RSC24760
         LA    WRKR2,0(WRKR2)          ZERO HIGH ORDER BYTE           RSC24770
         CR    WRKR2,TCBR                                             RSC24780
         BE    STM00031                                               RSC24790
         LA    WRKR1,4(WRKR1)          LOOK AT NEXT ENTRY             RSC24800
         B     STM00030                                               RSC24810
STM00031 ST    WRKR1,NXTTCB                                           RSC24820
         LA    WRKR1,ALCB1            PTR TO ADDR OF LCB1             RSC24830
STM00033 L     WRKR2,0(WRKR1)         ADDR OF LCB1                    RSC24840
         LA    WRKR2,0(WRKR2)         ZERO HIGH ORDER BYTE            RSC24850
         CR    WRKR2,LCBR             COMPARE ADDRESSES               RSC24860
         BE    STM00032               GOT A HIT                       RSC24870
         LA    WRKR1,4(WRKR1)         INCREMENT                       RSC24880
         B     STM00033                                               RSC24890
STM00032 ST    WRKR1,NXTLCB                                           RSC24900
         B     SCHDULR                 BRANCH                         RSC24910
ELLX2    DC    V(TCB1A)                                               RSC24920
         DC    V(TCB1B)                                               RSC24930
         EJECT                                                        RSC24940
********************************************************************* RSC24950
********************************************************************* RSC24960
**                                                                 ** RSC24970
**                         T S K A L O C                           ** RSC24980
**                                                                 ** RSC24990
********************************************************************* RSC25000
********************************************************************* RSC25010
**                                                                 ** RSC25020
**  CALLED BY.   MSGPROC.                                          ** RSC25030
**  ROUTINES CALLED.   RETURNS TO MSGPROC.                         ** RSC25040
**  CALLING CONVENTIONS.                                           ** RSC25050
**     1. NONE.                                                    ** RSC25060
**  FUNCTIONS.                                                     ** RSC25070
**     1. IF THE MAXIMUM NUMBER OF SUBTASKS ARE IN EXECUTION,      ** RSC25080
**        TSKALOC WILL WAIT UNTIL ONE OF THEM FINISHES, & THEN     ** RSC25090
**        WILL RETURN TO MSGPROC SO IT CAN CALL ANOTHER TERM       ** RSC25100
**        SUBROUTINE.                                              ** RSC25110
**     2. IF LESS THAN THE NUMBER OF SUBTASKS ARE IN EXECUTION,       ** RSC25120
**        TSKALOC IMMEDIATELY RETURNS CONTROL TO MSGPROC.             ** RSC25130
**                                                                    ** RSC25140
*********************************************************************** RSC25150
         SPACE 5                                                        RSC25160
TSKALOC  ST    LCBR,LCBRSW              SAVE LCBR AND TCBR              RSC25170
         ST    TCBR,TCBRSW                                              RSC25180
         SR    WRKR3,WRKR3              ZERO COUNTER                    RSC25190
         LA    SAVR1,TSKTBL             SET TASK TBL PTR TO FRST ENTRY  RSC25200
         LA    WRKR1,ALCB1              GET FIRST LINE                  RSC25210
TSKLND   L     LCBR,0(WRKR1)                                            RSC25220
         TM    LNSTAT,X'01'             IS LINE DOWN                    RSC25230
         BO    TSKNTL                   IF YES GO TO NEXT LINE          RSC25240
         LA    WRKR2,ATCB1              GET FIRST TERM ON LINE          RSC25250
TSKTMD   L     TCBR,0(WRKR2)                                            RSC25260
         TM    TSTAT,X'01'              IS TERM DOWN                    RSC25270
         BO    TSKNTT                   IF YES GO TO NEXT TERM          RSC25280
         TM    TECB,X'40'               IS TERM MSG PROCESSING          RSC25290
         BZ    TSKCNTR                  IF YES GO TO COUNTER            RSC25300
TSKNTT   TM    0(WRKR2),X'80'           IS THIS THE LAST TERM OF LINE   RSC25310
         BO    TSKNTL                   IF YES GO TO NEXT LINE          RSC25320
         LA    WRKR2,4(WRKR2)           GET NEXT TERM                   RSC25330
         B     TSKTMD                   GO TEST TERM                    RSC25340
TSKNTL   TM    0(WRKR1),X'80'           IS THIS THE LAST LINE           RSC25350
         BO    TSKEND                   IF YES GO FINISH UP             RSC25360
         LA    WRKR1,4(WRKR1)           GET NEXT LINE                   RSC25370
         B     TSKLND                   GO TEST LINE                    RSC25380
TSKCNTR  LA    SAVR2,TECB               GET ADDR OF TECB                RSC25390
         ST    SAVR2,0(SAVR1)           STORE TECB IN WAIT LIST         RSC25400
         LA    SAVR1,4(SAVR1)           UP LIST PTR TO NEXT ENTRY       RSC25410
         LA    WRKR3,1(WRKR3)            UP COUNTER BY ONE              RSC25420
         C     WRKR3,PARM2                                              RSC25430
         BNE   TSKNTT                   IF NO GO TO NEXT TERM           RSC25440
         S     SAVR1,=F'4'              BACK PTR UP BY ONE ENTY         RSC25450
         MVI   SAVR1,X'80'              SET EOL BIT                     RSC25460
         WAIT  1,ECBLIST=TSKTBL         WAIT UNTIL A TASK HAS FINISHED  RSC25470
```

```
TSKEND   L     LCBR,LCBRSW              RESTORE LCBR AND TCBR           RSC25480
         L     TCBR,TCBRSW                                              RSC25490
         B     RTNFTSK                                                  RSC25500
TCBRSW   DS    F                                                        RSC25510
LCBRSW   DS    F                                                        RSC25520
TSKTBL   DS    16F                                                      RSC25530
         EJECT                                                          RSC25540
*********************************************************************** RSC25550
*********************************************************************** RSC25560
**                                                                   ** RSC25570
**                           W R I T E                               ** RSC25580
**                                                                   ** RSC25590
*********************************************************************** RSC25600
*********************************************************************** RSC25610
**                                                                   ** RSC25620
**  CALLED BY.   OUTFRMT, SPCWRT, WTANS.                             ** RSC25630
**  ROUTINES CALLED.   SIOCK, SCHDULR.                               ** RSC25640
**  CALLING CONVENTIONS.                                             ** RSC25650
**     1. DECBR MUST POINT TO APPROPRIATE DECB OF LINE.              ** RSC25660
**     2. MSGARR MUST POINT TO SCPBF+4.                              ** RSC25670
**     3. LGTHR MUST CONTAIN LENGTH OF MESSAGE+1 (FOR ETX).          ** RSC25680
**     4. TRMLR MUST POINT TO APPROPRIATE POLLING LIST.              ** RSC25690
**  FUNCTIONS.                                                       ** RSC25700
**     1. ZERO DECB.                                                 ** RSC25710
**     2. ISSUE WRITE.                                               ** RSC25720
**     3. GIVE CONTROL BACK TO SCHDULR.                              ** RSC25730
**                                                                   ** RSC25740
*********************************************************************** RSC25750
         SPACE 5                                                        RSC25760
WRITE    MVC   DECSDECB,=F'0'          ZERO ECB IN DECB                 RSC25770
         ST    TCBR,TADECB             PUT ADDR OF TCB IN DECB          RSC25780
* ISSUE WRITE                                                           RSC25790
         WRITE (DECBR),TSR,,(MSGARR),(LGTHR),(TRMLR),,MF=E              RSC25800
         BAL   RETREG,SIOCK            CCW CHECK ROUTINE                RSC25810
         B     SCHDULR                 BRANCH                           RSC25820
         EJECT                                                          RSC25830
```

```
*********************************************************************** RSC25840
*********************************************************************** RSC25850
 *                                                                   ** RSC25860
**                         W R I N I T                               ** RSC25870
**                                                                   ** RSC25880
*********************************************************************** RSC25890
*********************************************************************** RSC25900
**                                                                   ** RSC25910
**  CALLED BY.   SCHDULR.                                            ** RSC25920
**  ROUTINES CALLED.   OUTFRMT.                                      ** RSC25930
**  CALLING CONVENTIONS.                                             ** RSC25940
**     1. TCBR MUST POINT TO TCB WHICH WILL RECEIVE INITIAL MES-     ** RSC25950
**        SAGE.                                                      ** RSC25960
**  FUNCTIONS.                                                       ** RSC25970
**     1. LOAD APPROPRIATE REGISTERS FOR OUTFRMT (MSGARR & LGTHR).   ** RSC25980
**     2. LOAD APPROPRIATE REGISTERS FOR WRITE (TRMLR & DECBR).      ** RSC25990
**     3. MOVE INITIAL MESSAGE INTO SCPBF+4 FOR TRANSLATION &        ** RSC26000
**        WRITE, 'BEGIN MEDICAL LIBRARY INQUIRY PROGRAM'.            ** RSC26010
**     4. STORE ERRCK IN NXTRN1, & MSGPROC IN NXTRN2.                ** RSC26020
**     5. LOAD ADDRESS OF WRITE ROUTINE INTO WRTREG.                 ** RSC26030
**     6. BRANCH TO OUTFRMT.                                         ** RSC26040
**                                                                   ** RSC26050
*********************************************************************** RSC26060
         SPACE 5                                                        RSC26070
WRINIT   L     DECBR,ADECB             LOAD ADDR OF DECB                RSC26080
         L     MSGARR,TSTAT            LOAD ADDR OF MSG AREA            RSC26090
         LA    MSGARR,4(MSGARR)        ALLOW FOR COUNT                  RSC26100
         MVC   0(IML,MSGARR),BEGINMSG  MOVE MSG INTO THEIR AREA         RSC26110
         LA    LGTHR,IML(0)            LOAD ADDR OF LENGTH              RSC26120
         L     TRMLR,TLST              LOAD ADDR OF TERMINAL LIST       RSC26130
         MVC   TPCW,=F'1'                                               RSC26140
         OI    TSTAT,X'02'             TURN ON BIT 6 OF IOS1            RSC26150
         NI    TSTAT,X'FB'             TURN OFF BIT 5 IN STATUS         RSC26160
         LA    WRKR1,ERRCK             PUT ADDR OF ERRCK IN NXTRN1      RSC26170
         ST    WRKR1,NXTRN1                                             RSC26180
         LA    WRKR1,MSGPROC           PUT ADDR OF MSGPROC IN NXTRN2    RSC26190
         ST    WRKR1,NXTRN2            TURN OFF BIT 6 OF BIOS2          RSC26200
         LA    WRTREG,WRITE            LOAD WRITE REGISTER              RSC26210
         B     OUTFRMT                 BRANCH TO OUTFORMAT              RSC26220
IML      EQU   41                      INITIAL MSG LENGTH               RSC26230
BEGINMSG DC    C'BEGIN MEDICAL LIBRARY INQUIRY PROGRAM...'              RSC26240
         DC    X'BD'                   START SYMBOL                     RSC26250
         EJECT                                                          RSC26260
************************************************************************RSC26270
************************************************************************RSC26280
**                                                                    **RSC26290
**                             W T A N S                              **RSC26300
**                                                                    **RSC26310
************************************************************************RSC26320
************************************************************************RSC26330
**                                                                    **RSC26340
**  CALLED BY.   SCHDULR.                                             **RSC26350
**  ROUTINES CALLED.   WRITE.                                         **RSC26360
**  CALLING CONVENTIONS.                                              **RSC26370
**      1. TCBR MUST CONTAIN ADDRESS OF TCB WHO IS GETTING THE REPLY  **RSC26380
**         MESSAGE TO A TERMINAL COMMAND THAT WAS ENTERED.            **RSC26390
**  FUNCTIONS.                                                        **RSC26400
**      1. LOAD REGISTERS NEEDED FOR WRITE ROUTINE (DECBR, MSGARR,    **RSC26410
**         LGTHR & TRMLR).                                            **RSC26420
**      2. STORE ADDRESS OF ERRCK INTO NXTRN1 & RESCHDLR INTO         **RSC26430
**         NXTRN2.                                                    **RSC26440
**      3. BRANCH TO WRITE ROUTINE.                                   **RSC26450
**                                                                    **RSC26460
************************************************************************RSC26470
         SPACE 5                                                        RSC26480
WTANS    L     DECBR,ADECB             LOAD ADDR OF DECB                RSC26490
         L     MSGARR,TSTAT            LOAD ADDR OF MESSAGE AREA        RSC26500
         LA    MSGARR,4(MSGARR)                                         RSC26510
         LA    LGTHR,ANSLGTH           LOAD LENGTH                      RSC26520
         L     TRMLR,TLST              LOAD ADDR OF TERMINAL LIST       RSC26530
         LA    WRKR1,ERRCK             PUT ADDR OF ERRCK IN NXTRN1      RSC26540
         ST    WRKR1,NXTRN1                                             RSC26550
         LA    WRKR1,RESCHDLR          PUT ADDR OF RESCHDLR IN NXTRN2   RSC26560
         ST    WRKR1,NXTRN2                                             RSC26570
         OI    BIOS2,X'02'                                              RSC26580
         B     WRITE                   BRANCH                           RSC26590
ANSLGTH  EQU   480                     MESSAGE LENGTH                   RSC26600
         EJECT                                                          RSC26610
SNXTTCB  DS    A                       ADDR FOR NXT TCB                 RSC26620
NRI      DC    16X'00'                NEGATIVE RESPNS INDIC - X'FF'     RSC26630
DCBADR   DC    V(DCBX)                                                  RSC26640
         SPACE 5                                                        RSC26650
         SPACE 3                                                        RSC26655
*                           ***********                                 RSC26660
*                           *         *                                 RSC26670
*                           *  S C B  *                                 RSC26680
*                           *         *                                 RSC26690
*                           ***********                                 RSC26700
SCB      DS    0F                                                       RSC26710
NXTLCB   DC    A(ALCB1)                                                 RSC26720
ALCB1    DC    V(LCB1)                                                  RSC26730
         SPACE 3                                                        RSC26740
         LTORG                                                          RSC26750
         EJECT                                                          RSC26760
LCBDS    DSECT                                                          RSC26770
         SPACE 3                                                        RSC26780
*        -------------------------------------------                    RSC26790
*        +                                         +                    RSC26800
*        +           ADDRESS OF NEXT TCB           +                    RSC26810
*        +                                         +                    RSC26820
*        +-----------------------------------------+                    RSC26830
*        +   LINE   +                              +                    RSC26840
*        +  STATUS  +        ADDRESS OF DECB       +                    RSC26850
*        +          +                              +                    RSC26860
*        +-----------------------------------------+                    RSC26870
*        +                                         +                    RSC26880
*        +             ADDRESS OF TCB1             +                    RSC26890
*        +                                         +                    RSC26900
*        +----------------------------------------+                       RSC26910
*        +                                        +                       RSC26920
*        +            ADDRESS OF TCB2             +                       RSC26930
*        +                                        +                       RSC26940
*        ------------------------------------------                       RSC26950
         SPACE 3                                                          RSC26960
LCB      DS    0F                                                         RSC26970
NXTTCB   DS    A                                                          RSC26980
LNSTAT   DS    0CL1                   LINE STATUS                         RSC26990
ADECB    DS    A                      ADDR OF DECB FOR THIS LINE          RSC27000
ATCB1    DS    A                      ADDR OF TCB1 FOR THIS LINE          RSC27010
ATCB2    DC    X'80'                  END OF LIST                         RSC27020
         DS    AL3                    ADDR OF TCB2 FOR THIS LINE          RSC27030
         EJECT                                                            RSC27040
TCBDS    DSECT                                                            RSC27050
         SPACE 3                                                          RSC27060
*        ------------------------------------------                       RSC27070
*        +                                        +                       RSC27080
*        +      TERMINAL EVENT CONTROL BLOCK      +                       RSC27090
*        +                                        +                       RSC27100
*        +----------------------------------------+                       RSC27110
*        +                                        +                       RSC27120
*        +    TERML PROCESS COMMUNICATION WORD    +                       RSC27130
*        +                                        +                       RSC27140
*        +----------------------------------------+                       RSC27150
*        +                                        +                       RSC27160
*        +       LENGTH OF OUTPUT MESSAGE         +                       RSC27170
*        +                                        +                       RSC27180
*        +----------------------------------------+                       RSC27190
*        +          +                             +                       RSC27200
*        +TERMINAL  +  ADDRESS OF MESSAGE AREA    +                       RSC27210
*        + STATUS   +                             +                       RSC27220
*        +----------------------------------------+                       RSC27230
*        +          +                             +                       RSC27240
*        +TERMINAL  +  ADDRESS OF TERMINAL LIST   +                       RSC27250
*        + NUMBER   +                             +                       RSC27260
*        +----------------------------------------+                     RSC27270
*        +                                        +                     RSC27280
*        +        ADDRESS OF NXTRN1               +                     RSC27290
*        +                                        +                     RSC27300
*        +----------------------------------------+                     RSC27310
*        +         +                              +                     RSC27320
*        + BIOS2   +     ADDRESS OF NXTRN2        +                     RSC27330
*        +         +                              +                     RSC27340
*        +----------------------------------------+                     RSC27350
*        +         +                              +                     RSC27360
*        +  LINE   +        ADDRESS OF LCB        +                     RSC27370
*        + NUMBER  +                              +                     RSC27380
*        +----------------------------------------+                     RSC27390
*        +                                        +                     RSC27400
*        +     ADDRESS OF SPECIFIC POLL AREA      +                     RSC27410
*        +                                        +                     RSC27420
*        +----------------------------------------+                     RSC27430
*        +                                        +                     RSC27440
*        + STANDARD EVENT CONTROL BLOCK FROM DECB +                     RSC27450
*        +                                        +                     RSC27460
*        +----------------------------------------+                     RSC27470
*        +                   +                    +                     RSC27480
*        +   DECB OP TYPE    +   LENGTH IN DECB   +                     RSC27490
*        +                   +                    +                     RSC27500
*        +----------------------------------------+                     RSC27510
*        +       DECB        +        DECB        +                     RSC27520
*        +    SENSE BYTES    +   RESIDUAL COUNT   +                     RSC27530
*        +    1    &    2    +                    +                     RSC27540
*        +----------------------------------------+                     RSC27550
*        +   DECB  +   DECB  +   DECB  +   DECB   +                     RSC27560
*        +  STATUS + RELATIVE+ ADDRESS + VRC/LRC  +                     RSC27570
*        +  FLAGS  + LINE NO.+ RESPONSE+ RESPONSE +                     RSC27580
*        +----------------------------------------+                     RSC27590
*        +  DECB   +  DECB   +        DECB        +                     RSC27600
*        +  TP-OP  +  ERROR  +     CSW STATUS     +                     RSC27610
*        +  CODE   +  STATUS +                    +                     RSC27620
*        -------------------------------------------                    RSC27630
         SPACE 5                                                        RSC27640
*********************************************************************** RSC27650
* TERMINAL STATUS BYTE                                                * RSC27660
*                                                                     * RSC27670
*        BITS 0 THRU 3        NOT USED                                * RSC27680
*        BIT  4               JUST FINISHED MSG PROCESSING BIT        * RSC27690
*        BIT  5               LAST I/O OPERATION FOR TERM WAS A WRITE * RSC27700
*        BIT  6               IOSI BIT - ON MEANS I/O, OFF MEANS PROC * RSC27710
*        BIT  7               TERMINAL DROPPED BIT                    * RSC27720
*                                                                     * RSC27730
*********************************************************************** RSC27740
* TERMINAL PROCESSING COMMUNICATION WORD - TPCW                       * RSC27750
*                                                                     * RSC27760
*        BITS 0 THRU 26       NOT USED                                * RSC27770
*        BIT  27              PROCESS BIT                             * RSC27780
*        BIT  28              ERASE/WRITE FULL BUFFER BIT             * RSC27790
*        BIT  29              READ MANUAL INPUT BIT                   * RSC27800
*        BIT  30              READ FULL BUFFER BIT                    * RSC27810
*        BIT  31              /STOP BIT                               * RSC27820
*                                                                     * RSC27830
*********************************************************************** RSC27840
         SPACE 3                                                        RSC27850
TCB      DS    0F                                                       RSC27860
TECB     DS    CL1                      EVENT CTRL BLK FOR TERMINAL     RSC27870
         DS    CL3                                                      RSC27880
TPCW     DS    F                        TERM PROCESS COMMUNICATION WORD RSC27890
TML      DS    F                        TERM MSG LENGTH                 RSC27900
TSTAT    DS    CL1                      TERM STATUS                     RSC27910
TMA      DS    AL3                      TERM MSG AREA                   RSC27920
TNO      DS    0CL1                     TERM NUMBER                     RSC27930
TLST     DS    A                        ADDR OF TERMINAL LIST           RSC27940
NXTRN1   DS    A                        ADDR OF NEXT RTN 1              RSC27950
BIOS2    DS    0CL1                     BYTE IOS2                       RSC27960
NXTRN2   DS    A                        ADDR OF NEXT RN2 2              RSC27970
TLNO     DS    0CL1                                                     RSC27980
```

```
TLCB     DS    A                            ADDR OF LCB                     RSC27990
TSPAREA  DS    A                            ADDR OF AREA FOR SPECIFIC POLL  RSC28000
TSECB    DS    F                            STD EVENT CTRL BLK FROM DECB    RSC28010
TOPTYP   DS    H                            OP TYPE                         RSC28020
TDECLNTH DS    H                            LENGTH IN DECB                  RSC28030
TSNS1    DS    CL1                          SENSE BYTE 1                    RSC28040
TSNS2    DS    CL1                          SENSE BYTE 2                    RSC28050
TRSDCNT  DS    H                            RESID COUNT                     RSC28060
TSTSFLGS DS    CL1                          STATUS FLAGS                    RSC28070
TRLNO    DS    CL1                          RELATIVE LINE NO                RSC28080
TADRSPNS DS    CL1                          RESPONSE TO ADDRESSING          RSC28090
TVRCLRC  DS    CL1                          RESPONSE TO VRC/LRC             RSC28100
TTOPC    DS    CL1                          TP-OP CODE                      RSC28110
TERRSTS  DS    CL1                          ERROR STATUS                    RSC28120
TCSW     DS    H                            CSW STATUS                      RSC28130
         EJECT                                                              RSC28140
* DEFINE DECB DUMMY SECTION                                                 RSC28150
         IECTDECB                                                           RSC28160
TADECB   DS    A                                                            RSC28170
LADECB   DS    A                                                            RSC28180
         END                                                                RSC28190
```

```
TABLES    START 0
           ENTRY IECTRSCI
          ENTRY IECTSSCI
          ENTRY TRTSTBL
IECTRSCI  EQU    *
*                  0 1 2 3 4 5 6 7 8 9 A B C D E F
          DC     16X'40'                                0
          DC     16X'40'                                1
          DC     16X'40'                                2
          DC     16X'40'                                3
          DC     X'4040407B5B6C507D4D5D5C4E6B604B61'    4
          DC     X'F0F1F2F3F4F5F6F7F8F97A5E4C7E6E6F'    5
          DC     16X'40'                                6
          DC     16X'40'                                7
          DC     16X'40'                                8
          DC     16X'40'                                9
          DC     X'40C1C2C3C4C5C6C7C8C9D1D2D3D4D5D6'    A
          DC     X'D7D8D9E2E3E4E5E6E7E8E9404040406D'    B
          DC     16X'40'                                C
          DC     16X'40'                                D
          DC     X'7C404040404040404040404040404040'    E
          DC     X'4040404040404040404040405F404F40'    F
IECTSSCI  EQU    *
          DC     X'40404040404040404040400A40404040'    0
          DC     16X'40'                                1
          DC     16X'40'                                2
          DC     16X'40'                                3
          DC     X'40404040404040404040404E5C484BFE'    4
          DC     X'4640404040404040404040444A495BFC'    5
          DC     X'4D4F404040404040404040404C45BF5E5F'  6
          DC     X'404040404040404040405A43E0475D40'    7
          DC     X'40A1A2A3A4A5A6A7A8A9404040404040'    8
          DC     X'40AAABACADAEAFB0B1B2404040404040'    9
          DC     X'4040B3B4B5B6B7B8B9BA404040404040'    A
          DC     X'40404040404040404040404040BD4040'    B
          DC     X'40A1A2A3A4A5A6A7A8A9404040404040'    C
```

```
         DC    X'40AAABACADAEAFB0B1B2404040404040'      D
         DC    X'4040B3B4B5B6B7B8B9BA404040404040'      E
         DC    X'505152535455565758595A4040404040'      F
TRTSTBL  EQU   *
         DC    X'000000FF'
         DC    63F'0'
         END
```

```
CBLKSECT START 0
         ENTRY DCBX
         ENTRY LCB1
         ENTRY TCB1A,TCB1B
         ENTRY DECB1
         SPACE 5
* SET UP LINE ERROR BLOCK FOR STORAGE AREA
LER      LERB  1
* DEFINITION OF DCB
DCBX     DCB   DSORG=CX,MACRF=(R,W),DDNAME=LINEX,LERB=LER,ERROPT=RWC
         SPACE 5
* LINE CONTROL BLOCK                    WORD
LCB1     DC    A(LCB1+8)                  1.NEXT TCB
         DC    X'00'                      2.LINE STATUS
         DC    AL3(DECB1)                  .ADDR OF DECB
         DC    A(TCB1A)                   3.ADDR OF TCB 1
         DC    X'80'                      4.FLAG POINTING TO LAST LCB
         DC    AL3(TCB1B)                  .ADDR OF TCB 2
         DS    0F
         SPACE 5
* TERMINAL CONTROL BLOCKS               WORD
* TERM 1
TCB1A    DC    X'40000000'                1.EVENT CONTROL BLOCK FOR TCB
         DC    X'00000001'                2.TERMINAL PROC CONTROL WORD
         DC    F'480'                     3.LENGTH
         DS    0C                         4.TERMINAL STATUS BITS
         DC    VL4(SCPBF1)                 .ADDR OF MESSAGE AREA
         DC    X'00'                      5.PHYSICAL TERMINAL NUMBER
         DC    AL3(TLST1A)                 .ADDR OF TERMINAL LIST
         DC    V(WRINIT)                  6.NEXT ROUTINE 1
         DC    X'00'                      7.IO STATUS 2
         DC    X'000000'                   .NEXT ROUTINE 2
         DC    X'00'
         DC    AL3(LCB1)
         DC    VL4(SCPBF1)
* WORDS 9-13 ARE SAVED FROM THE DECB
```

```
         DC    F'0'                      9.STANDARD EVENT CONTROL BLOCK
         DC    H'0'                     10.OPERATION TYPE
         DC    H'0'                       .MESSAGE AREA LENGTH
         DC    X'00'                    11.1ST SENSE BYTE
         DC    X'00'                      .2ND SENSE BYTE
         DC    H'0'                       .RESIDUAL COUNT
         DC    X'00'                    12.STATUS FLAGS
         DC    X'00'                      .RELATIVE LINE NUMBER
         DC    X'00'                      .RESPONSE TO ADDRESSING
         DC    X'00'                      .RESPONSE TO VRC/LRC
         DC    X'00'                    13.TP-OP CODE
         DC    X'00'                      .ERROR STATUS
         DC    H'0'                       .CSW STATUS
         SPACE 5
* TERM 2
TCB1B    DC    X'40000000'               1.EVENT CONTROL BLOCK FOR TCB
         DC    X'00000001'               2.TERMINAL PROC CONTROL WORD
         DC    F'480'                    3.LENGTH
         DS    0C                        4.TERMINAL STATUS BITS
         DC    VL4(SCPBF2)                .ADDR OF MESSAGE AREA
         DC    X'01'                     5.PHYSICAL TERMINAL NUMBER
         DC    AL3(TLST1B)                .ADDR OF TERMINAL LIST
         DC    V(WRINIT)                 6.NEXT ROUTINE 1
         DC    X'00'                     7.IO STATUS
         DC    X'000000'                  .NEXT ROUTINE 2
         DC    X'00'
         DC    AL3(LCB1)
         DC    VL4(SCPBF2)
* WORDS 9-13 ARE SAVED FROM THE DECB
         DC    F'0'                      9.STANDARD EVENT CONTROL BLOCK
         DC    H'0'                     10.OPERATION TYPE
         DC    H'0'                       .MESSAGE AREA LENGTH
         DC    X'00'                    11.1ST SENSE BYTE
         DC    X'00'                      .2ND SENSE BYTE
         DC    H'0'                       .RESIDUAL COUNT
         DC    X'00'                    12.STATUS FLAGS
         DC    X'00'                       .RELATIVE LINE NUMBER
         DC    X'00'                       .RESPONSE TO ADDRESSING
         DC    X'00'                       .RESPONSE TO VRC/LRC
         DC    X'00'                     13.TP-OP CODE
         DC    X'00'                       .ERROR STATUS
         DC    H'0'                        .CSW STATUS
         DC    X'FFFFFFFF'                FLAG DEFINING END OF TCB'S
* FUTURE TCB ADDITIONS SHOULD BE INSERTED HERE
         SPACE 5
TLST1A   DFTRMLST OPENLST,(A0A0)
TLST1B   DFTRMLST OPENLST,(A0A1)
         SPACE 5
* SET UP DECB
         READ  DECB1,T,DCBX,,,,1,MF=L
         DC    A(TCB1A)                   ADDR OF TCB
         DC    A(LCB1)                    ADDRESS OF LCB
         END
         MACRO                                                          TERM0000
&TAG     MONITOR &TRM,&MODNM=,&ERRMOD=,&MODS=                           TERM0010
          GBLA  &TNO,&ERRTIN                                            TERM0020
          GBLC  &MDNAM                                                  TERM0030
          LCLA  &A,&MODNO(100),&I,&J,&K,&L,&S1,&S2,&S3,&S4,&ID,&N       TERM0040
          LCLC  &X,&STRT,&FNX,&M,&Z,&ZERO                               TERM0050
&A        SETA  &SYSNDX                                                 TERM0060
&ID       SETA  &TNO+1                                                  TERM0070
          AIF   (T'&TRM EQ 'O').MYNO                                    TERM0080
&ID       SETA  &TRM                                                    TERM0090
.MYNO     ANOP                                                          TERM0100
&TNO      SETA  &ID                                                     TERM0110
          AIF   (T'&MODNM EQ 'O').MYNM                                  TERM0120
&MDNAM    SETC  '&MODNM'                                                TERM0130
.MYNM     AIF   ('&MDNAM' NE '').ERNO                                   TERM0140
&MDNAM    SETC  'MOD'
.ERNO     AIF   (T'&ERRMOD EQ 'O').MYER                                 TERM0160
&ERRTIN   SETA  &ERRMOD                                                 TERM0170
.MYER     AIF   (&ERRTIN NE 0).GO                                       TERM0180
&ERRTIN   SETA  99                                                      TERM0190
.GO       AIF   (&TNO NE 1).NTL                                         TERM0200
          TITLE 'TERMINAL MONITOR                    WW JONES + E S DAHL' TERM0210
.NTL      ANOP                                                          TERM0220
TERM&ID   CSECT                                                         TERM0230
          SPACE                                                         TERM0240
*        TERMX    UNM-MEDLIBY LEVEL=68253                               TERM0250
*                 PATCHES=0                                             TERM0260
          SPACE 5                                                       TERM0270
*       * START HOUSEKEEPING                                            TERM0280
          SAVE  (14,12),,* ENTRY POINT FROM IO2260                      TERM0290
          BALR  12,0                                                    TERM0300
          USING *,12                                                    TERM0310
*        STORE CALL CHAINING POINTERS                                   TERM0320
          ST    13,IJSVE&A+4 SAVE BACK POINTER IN NEW SAVE AREA         TERM0330
          LR    11,13                                                   TERM0340
          LA    13,IJSVE&A LOAD 13 FOR SUBSEQUENT CALLS                 TERM0350
         ST    13,8(11) LOAD FORWARD POINTER IN OLD SAVE AREA           TERM0360
*       RESOLVE PARAM LIST                                              TERM0370
         L     8,0(1) GET ECB ADDR IN REG 8                             TERM0380
*     * END   HOUSEKEEPING                                              TERM0390
         SPACE 5                                                        TERM0400
*     * START ENTRY VECTOR                                              TERM0410
         TM    7(8),1                                                   TERM0420
         BO    STPDRP&ID IF STOP/DROP BIT IS ON                         TERM0430
         TM    7(8),19                                                  TERM0440
         BM    LOOKUP&ID IF READ (MI/FB) OR PROCESS BIT ON IN TPW       TERM0450
         TM    SWIT&ID,3                                                TERM0460
         BZ    LOOKUP&ID IF NO RDNXT (MI/FB) ON IN TCM                  TERM0470
*     * END   ENTRY VECTOR                                              TERM0480
         SPACE 5                                                        TERM0490
*     * START BUFFER INITIALIZATON                                      TERM0500
         LA    9,SCPBF&ID FIND THE SCOPE BUFFER                         TERM0510
         MVI   4(9),64 SET THE CLEARING CHARACTER                       TERM0520
         MVC   5(240,9),4(9) CASCADE THE FRST STRING                    TERM0530
         MVC   245(240,9),244(9) AND THE NEXT STRING                    TERM0540
*     * END    INITIALIZOR                                              TERM0550
         EJECT                                                          TERM0560
         SPACE 9                                                        TERM0570
*     * START EXIT SWITCHING                                            TERM0580
         IC    9,SWIT&ID GET NEXT OP                                    TERM0590
         SLL   9,1(0) MPY BY TWO                                        TERM0600
         STC   9,7(8) STORE IN TPW                                      TERM0610
STWTNX&ID MVI  SWIT&ID,4 SET WRITE-NEXT                                 TERM0620
IJPST&A  POST  (8)                                                      TERM0630
         L     13,IJSVE&A+4 FIND OLD SAVE AREA                          TERM0640
        RETURN (14,12) TO IO2260                                        TERM0650
*     * END   EXIT PROCESSING                                           TERM0660
         SPACE 15                                                       TERM0665
*     * START STOP/DROP HANDLING                                        TERM0670
STPDRP&ID MVI  7(8),4 SET TPW TO RD(MI)                                 TERM0680
         XC    SWIT&ID+1(3),SWIT&ID+1 SET MODULE NUMBER = 0             TERM0690
         TM    SWIT&ID,8                                                TERM0700
         BO    STWTNX&ID IF IT WAS FIRST TIME                           TERM0710
         MVI   SWIT&ID,8 SET FIRSTIME FOR ERROR                         TERM0720
         B     IJPST&A POST & GO BACK                                   TERM0730
*      * END   STOP/DROP                                                TERM0740
         EJECT                                                          TERM0750
         SPACE 12                                                       TERM0760
*      * START LOOKUPS FOR NEXT ATTACH                                  TERM0770
LOOKUP&ID LH   9,SWIT&ID+2 GET NEXT MODULE NO IN REG 9                  TERM0780
         LTR   9,9                                                      TERM0790
         BC    7,GETMOD&ID IF MODNO NOT 0                               TERM0800
         SPACE 5                                                        TERM0810
*      * START ROUTINE TO FIND LIBRARIAN FUNCTION                       TERM0820
         LA    3,LIBLST&ID-8                                            TERM0830
         LA    5,ENDLIB&ID                                              TERM0840
         LA    4,8                                                      TERM0850
LIBLUP&ID BXH  3,4,GETMOD&ID WHEN TABLE IS EXHAUSTED                    TERM0860
         CLC   SCPBF&ID+4(6),0(3) WAS A LIBRARIAN FUNCTION REQUESTED    TERM0870
         BNE   LIBLUP&ID                                                TERM0880
         LH    9,6(3) GET SUCCESSFUL MODNO                              TERM0890
         BCTR  9,0 RELATIVE ZERO                                        TERM0892
         SPACE 5                                                        TERM0900
*      * START ROUTINE TO SELECT REQUESTED MODULE                       TERM0910
GETMOD&ID SLL  9,3(0) TO MULTIPLY BY 8 FOR INDEX                        TERM0920
         LA    9,MODLST&ID.(9) SET REQUESTED MODNAME ADDR IN R9         TERM0930
         LA    10,ENDLST&ID TO SET UPPER LIMIT                          TERM0940
         CR    9,10 TO VALIDATE REQUEST                                 TERM0950
         BL    *+6 IF WITHIN BOUNDS                                     TERM0960
         LR    9,10 ELSE SET FOR ERROR MODULE                           TERM0970
*      * END   LOOKUP PROCEDURES                                        TERM0980
         EJECT                                                          TERM0990
         SPACE 12                                                       TERM1000
       ATTACH EPLOC=(9),                                               1TERM1010
               PARAM=((8),RTLIS&A,MSG&A,SCPBF&ID,DABUF&A,SWIT&ID),     2TERM1020
               ECB=(8)                                                  TERM1030
         SPACE                                                          TERM1040
         B     IJPST&A . . . POST AND GO BACK                           TERM1050
```

```
               EJECT                                                  TERM1060
.* TO CREATE THE MODULE NAME TABLE                                    TERM1070
.*                                                                    TERM1080
.BLD           ANOP                                                   TERM1090
&I             SETA  &I+1                                             TERM1100
&MODNO(&I)     SETA  &ERRTIN                                          TERM1110
               AIF   (&I LT 100).BLD                                  TERM1120
&I             SETA  &I-&I                                            TERM1130
.STRT          AIF   (&I EQ N'&MODS).NDTBL                            TERM1140
&J             SETA  0                                                TERM1150
&I             SETA  &I+1                                             TERM1160
.SCAN          AIF   (K'&MODS(&I) NE &J).SCN                          TERM1170
               AIF   (&MODS(&I) GT 100).STRT                          TERM1180
&K             SETA  &MODS(&I)                                        TERM1190
&MODNO(&K)     SETA  &MODS(&I)                                        TERM1200
               AGO   .STRT                                            TERM1210
.SCN           ANOP                                                   TERM1220
&J             SETA  &J+1                                             TERM1230
               AIF   ('&MODS(&I)'(&J,1) NE '-').SCAN                  TERM1240
&L             SETA  K'&MODS(&I)                                      TERM1250
&ZERO          SETC  '00000000'                                       TERM1260
&STRT          SETC  '&ZERO'(1,9-&J).'&MODS(&I)'                      TERM1270
&FNX           SETC  '&ZERO'(1,8-(&L-&J)).'&MODS(&I)'(&J+1,&L-&J)     TERM1280
&J             SETA  &STRT                                            TERM1290
&K             SETA  &FNX                                             TERM1300
&L             SETA  1                                                TERM1310
               AIF   (&J LT &K).POS                                   TERM1320
&L             SETA  0-1                                              TERM1330
.POS           ANOP                                                   TERM1340
               AIF   (&J GT 100).NOP                                  TERM1350
&MODNO(&J)     SETA  &J                                               TERM1360
.NOP           ANOP                                                   TERM1370
               AIF   (&J EQ &K).STRT                                  TERM1380
&J             SETA  &J+&L                                            TERM1390
               AGO   .POS                                             TERM1400
.NDTBL         ANOP                                                   TERM1410
```

```
&K         SETA  100                                                       TERM1420
.FNDND     AIF   (&MODNO(&K) NE &ERRTIN).STP                               TERM1430
&K         SETA  &K-1                                                      TERM1440
           AIF   (&K NE 1).FNDND                                           TERM1450
.STP       ANOP                                                            TERM1460
           AIF   (&K GE 56).NSP                                            TERM1470
&N         SETA  (56-&K)/2                                                 TERM1480
.NSP       ANOP                                                            TERM1490
*        DEFINE TABLES                                                     TERM1500
           SPACE &N                                                        TERM1510
*      * START VALID MODULE NAMES                                          TERM1520
&X         SETC  'MODLST'.'&ID'                                            TERM1520
&J         SETA  1                                                         TERM1530
.PUT       ANOP                                                            TERM1560
           AIF   (&J GT &K).DID                                            TERM1570
&X         DC    CL8'&MDNAM.&MODNO(&J)'                                    TERM1580
&X         SETC  '&Z'                                                      TERM1590
&J         SETA  &J+1                                                      TERM1600
           AGO   .PUT                                                      TERM1610
.DID       ANOP                                                            TERM1620
ENDLST&ID  DC   CL8'&MDNAM.&ERRTIN' MUST BE LAST DC STATEMENT IN MODLST TERM1630
           EJECT                                                           TERM1640
           SPACE 9                                                         TERM1650
*      * START LIBRARIAN FUNCTIONS                                         TERM1660
LIBLST&ID  DC    C'ORDER '                                                 TERM1670
           DC    H'50'                                                     TERM1680
           DC    C'RECEIV'                                                 TERM1690
           DC    H'60'                                                     TERM1700
           DC    C'CATALO'                                                 TERM1710
           DC    H'70'                                                     TERM1720
           DC    C'INQUIR'                                                 TERM1730
           DC    H'90'                                                     TERM1740
ENDLIB&ID  DC    C'      '                                                 TERM1780
           DC    H'&ERRTIN' -ERROR- ID = SPACES                            TERM1790
           SPACE 10                                                        TERM1800
*        DEFINE STORAGE                                                    TERM1810
```

```
MSG&A        DS     120F UNIVERSAL COMMUNICATION AREA                   TERM1820
SCPBF&ID     DS     122F 2260 BUFFER WITH LEADING LENGTH WORD           TERM1830
DABUF&A      DS     131F DASD I/O AREA                                  TERM1840
IJSVE&A      DS     18F  NEW SAVE AREA                                  TERM1850
             SPACE 5                                                    TERM1860
*         DEFINE CONSTANTS                                              TERM1870
SWIT&ID      DC     X'08000000' THE TASK CONTROL MONITOR (TCM)          TERM1880
RTLIS&A      DC     V(RTLIST)                                           TERM1890
&FNX         SETC   '&TAG'                                              TERM1891
             AIF    ('&TAG' NE '').TAG                                  TERM1892
&FNX         SETC   'TERM'.'&ID'                                        TERM1893
.TAG         AIF    ('&FNX' EQ 'TERM'.'&ID').NTRY                       TERM1894
&FNX         EQU    TERM&ID                                             TERM1900
.NTRY        ENTRY SCPBF&ID,&TAG                                        TERM2000
*         * END    TERM-X                                               TERM2010
             EJECT                                                      TERM2020
          MEND                                                          TERM2020
TOA0     MONITOR 1,MODS=(1-20,50-56,60,61,70-80,87-90,81-86,91-96,57)
          EJECT
TOA1     MONITOR 2,MODS=(1-20,50-56,60,61,70-80,87-90,81-86,91-96,57)
             END
```

```
IONQ     TITLE 'I/O INTERFACE PROGRAM -IDENQ- BOB BOTTS - ED S DAHL'     IONQ  10
IDENQ    START 0                                                         IONQ  20
         SAVE  (14,12),,* '                                              IONQ  30
         BALR  12,0                      BASE 12                         IONQ  40
         USING *,12                                                      IONQ  50
         ST    13,SA+4                                                   IONQ  60
         LR    11,13                                                     IONQ  70
         LA    13,SA                                                     IONQ  80
         ST    13,8(11)                                                  IONQ  90
         SPACE 5                                                         IONQ 100
         L     2,0(1)                    2 PTS TO RLDARN                 IONQ 110
         L     3,4(1)                    3 PTS TO ADDR OF RTLIST         IONQ 120
         L     3,0(3)                    R3 PTS TO RLIST                 IONQ 130
         LA    4,8(1)                    ADD 8 TO 1 AND STORE IN R8      IONQ 140
         L     2,0(2)                    2 CONTAINS REL DA RTN NO.       IONQ 150
         LA    6,RNAME(5)                R6 PTS TO RNAME                 IONQ 160
         ENQ   (QNAME,(6),E,2,STEP),RET=HAVE                             IONQ0170
         LA    1,NUPRMLST R1 HAS V(NEW PARAM LIST)                       IONQ 290
         LA    7,CNTRLBLK GET DISPL OF CONTROL BLOCK POINTER             IONQ 300
         SLL   7,2(0) MPY BY PTR LEN (4)                                 IONQ 310
         L     8,0(3,7) GET ADDR OF CTL BLK PTR                          IONQ 320
         MVC   0(8,1),0(4) PICKUP DIRA AND KEY PTRS                      IONQ 330
         ST    8,8(1) PICKUP CTL BLK PTR                                 IONQ 340
         OI    8(1),X'80' ULTIMATE VARYLIST BIT                          IONQ 350
         CLC   0(2,6),RNAME+(16*2)                                       IONQ 360
         BL    NUPRMLST+12 IF LOWER THAN RDSCRT                          IONQ 370
         CLC   0(2,6),RNAME+(17*2)                                       IONQ 380
         BL    NUPRMLST+12 IF HIGHER THAN WTSCRT                         IONQ 390
         B     NUPRMLST+16                                               IONQ 400
         CNOP  0,4                                                       IONQ 410
NUPRMLST DC    3A(0)                                                     IONQ 420
         NI    4(1),X'7F' SET INNER VARYLIST BIT                         IONQ 430
         L     15,0(2,3)                 15 CONTAINS A(DA IO RTN)        IONQ 440
         BALR  14,15                     CALL                            IONQ 450
         EJECT                                                           IONQ 460
RESET13  L     13,4(13) CHAIN BACK ONE SAVE AREA                         IONQ0452
```

```
         L     5,0(4) FIND D A BUFFER                                   IONQ0454
         CLI   0(5),X'BC' WAS IT A COBOL NODEQU REQST                   IONQ0456
         BE    TSTERR IF IT WAS                                         IONQ0458
         CLI   0(5),X'BF' WAS IT A FORTRAN NODEQU REQST                 IONQ0460
         BNE   DEQUEU IF IT WASN''T                                     IONQ0462
         LH    9,2(5) GET RECORD LENGTH                                 IONQ0464
         EX    9,MVCFORT                                                IONQ0466
         B     RTRN                                                     IONQ0468
MVCFORT  MVC   0(0,5),4(5) SHIFT RECORD LEFT 1 WORD                     IONQ0469
DEQUEU   DEQ   (QNAME,(6),2,STEP)                                       IONQ0470
TSTERR   CLI   0(5),X'BD' TEST FOR I/O ERROR                            IONQ0590
         BE    LOOKBACK+4                                               IONQ0600
RTRN     RETURN (14,12)                                                 IONQ 630
         EJECT                                                          IONQ 640
SA       DC    18A(0)                                                   IONQ 650
QNAME    DC    C'DAIORTNE'                                              IONQ 660
RNAME    DC    C'00'                                                    IONQ 670
         DC    C'01'                                                    IONQ 680
         DC    C'02'                                                    IONQ 690
         DC    C'03'                                                    IONQ 700
         DC    C'04'                                                    IONQ 710
         DC    C'05'                                                    IONQ 720
         DC    C'06'                                                    IONQ 730
         DC    C'07'                                                    IONQ 740
         DC    C'08'                                                    IONQ 750
         DC    C'09'                                                    IONQ 760
         DC    C'10'                                                    IONQ 770
         DC    C'11'                                                    IONQ 780
         DC    C'12'                                                    IONQ 790
         DC    C'13'                                                    IONQ 800
         DC    C'14'                                                    IONQ 810
         DC    C'15'                                                    IONQ 820
         DC    C'16'                                                    IONQ 830
         DC    C'17'                                                    IONQ 840
         DC    C'18'                                                    IONQ 850
         DC    C'19'                                                    IONQ 860
         DC    C'20'                                                    IONQ 870
         DC    C'21'                                                    IONQ 880
         DC    C'22'                                                    IONQ 890
         DC    C'23'                                                    IONQ 900
         DC    C'24'                                                    IONQ 910
         DC    C'25'                                                    IONQ 920
         SPACE                                                          IONQ 930
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * IONQ 940
FILELIST DC    F'18'   RELATIVE LOCATION OF FLENMLST IN IOTABLE * * * * IONQ 950
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * IONQ 960
         SPACE                                                          IONQ 970
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * IONQ 980
CNTRLBLK DC    F'19'   RELATIVE LOCATION OF CONTROL BLOCK IN IOTABLE* * IONQ 990
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * IONQ1000
         EJECT                                                          IONQ1010
* BEGIN *ERROR*PROCESSING ROUTINE                                       IONQ1020
         SPACE                                                          IONQ1030
*      * START ROUTINE TO FIND TERMINAL MONITOR                         IONQ1040
LOOKBACK L     13,4(13) CHAIN BACK TO NEXT PREVIOUS SAVE AREA           IONQ1050
         L     15,16(13) GET ENTRY POINT NEXT SUBSEQUENT CALL           IONQ1060
         CLC   5(4,15),=C'TERM'                                         IONQ1070
         BNE   LOOKBACK IF NOT TERMINAL MONITOR                         IONQ1080
         L     13,8(13) CHAIN FORWARD ONE SAVE AREA                     IONQ1090
         LM    8,9,52(13) GET ECB AND MODNAME POINTERS                  IONQ1100
         CLC   0(3,9),=C'MOD' DOUBLE CHECK                              IONQ1110
         BNE   LOOKBACK 'TERM' WAS A COINCIDENCE                        IONQ1120
*        RESOLVE PARAMETERS                                             IONQ1130
         L     1,24(13) GET MODLIST POINTER                             IONQ1140
         L     2,08(1)  SCOPE BUFFER ADDR                               IONQ1150
         L     3,20(1)  TASK MONITOR ADDR                               IONQ1160
*       RESET  MONITOR                                                  IONQ1170
         MVC   4(4,8),=F'8' SET TPW=WRITE                               IONQ1180
         MVC   0(4,3),=F'99' NEXT MOD NO                                IONQ1190
         MVI   0(3),X'08' TCM=WRITE-NEXT                                IONQ1200
*        RESOLVE PARAMETERS                                             IONQ1210
         S     4,=F'8'  GET I/O LIST POINTER                            IONQ1220
         L     5,0(4)    GET RLDARN POINTER                              IONQ1230
         L     5,0(5)    GET RLDARN VALUE                                IONQ1240
         L     6,FILELIST DISPL(1) OF FLENMLST IN IOTABL                 IONQ1250
         SLL   6,2(0)    MPY BY 4                                        IONQ1260
         L     7,4(1)    GET IORT LIST PTR                               IONQ1270
         A     6,0(7) 6=FILE LIST ADDR                                   IONQ1280
         L     6,0(6)    FLNMLST PTR                                     IONQ1290
         SLL   5,3(0)    MPY BY ENTRY LEN                                IONQ1300
         AR    6,5        INDX TO FILE NAME                              IONQ1310
         L     4,12(4)   GET KEY ADDR                                    IONQ1320
*        INITLIZE SCOPE PROCESSING                                       IONQ1330
         MVI   4(2),X'40'        ) CLEAR                                 IONQ1340
         MVC   5(240,2),4(2)     ) SCOPE                                 IONQ1350
         MVC   245(240,2),244(2) BUFFER                                  IONQ1360
         LA    3,4(2) SET LINE POINTER                                   IONQ1370
         LR    2,3      TEXT AREA ADDRESS                                IONQ1380
*        BUILD MESSAGE                                                   IONQ1390
         MVC   0(38,3),=C'REPORT THE FOLLOWING DATA TO LIBRARIAN'        IONQ1400
         LA    3,80(3) SKIP A LINE                                       IONQ1410
         MVC   0(33,3),=C'DASD I/O ERROR HAS OCCURRED USING'             IONQ1420
         MVC   40(24,3),=C'THE FOLLOWING PARAMETERS'                     IONQ1430
         LA    3,80(3) NEXT LINE                                         IONQ1440
         MVC   0(23,3),=C'TERMINAL=       ,MODULE='                      IONQ1450
         IC    8,4(0,15) GET LENGTH OF MONITOR NAME                      IONQ1460
         BCTR  8,0 REL -0-                                               IONQ1470
         EX    8,*+4                                                     IONQ1480
         MVC   9(0,3),5(15) GET MONITOR NAME                             IONQ1490
         CLI   14(3),X'40'                                               IONQ1500
         BNE   *+14                                                      IONQ1510
         MVC   14(1,3),13(3)                                             IONQ1520
         MVI   13(3),X'F0'                                               IONQ1530
         MVC   23(6,3),0(9) GET MODULE NAME                              IONQ1540
         LA    3,40(3) NEXT LINE                                         IONQ1550
         EJECT                                                           IONQ1560
*      * MAIN BODY CONSTRUCTION                                          IONQ1570
*        FILE NAME HANDLER                                               IONQ1580
```

```
         MVC   0(5,3),=C'FILE='                                         IONQ1590
         LA    9,5(3) CHAR PTR                                          IONQ1600
         LA    10,1 SET INCREMENT                                       IONQ1610
         LA    11,6(6) SET LIMIT                                        IONQ1620
MOVENAME CLI   0(6),X'40'                                               IONQ1630
         BE    BXLENAME IF A SHORT NAME                                 IONQ1640
         MVC   0(1,9),0(6) TRANSFER FILENAME                            IONQ1650
         LA    9,1(9) BUMP THE CHAR PTR                                 IONQ1660
BXLENAME BXLE  6,10,MOVENAME                                            IONQ1670
*        KEY TEXT HANDLER                                               IONQ1680
         MVC   0(5,9),=C',KEY='                                         IONQ1690
         LA    9,5(9) ADVANCE CHAR PTR                                  IONQ1700
         LA    11,38(3) SET 11 TO EOL                                   IONQ1710
         LA    10,1                                                     IONQ1720
         XR    7,7                                                      IONQ1730
         IC    7,0(6) GET KEY LEN                                       IONQ1740
         LTR   7,7                                                      IONQ1750
         BZ    FORTKEY IF A FORTRAN FILE                                IONQ1760
SETKEY   AR    7,4     7=EOD                                            IONQ1770
         LR    6,10                                                     IONQ1780
MOVEKEY  MVC   0(1,9),0(4) TRANSFER KEY                                 IONQ1790
         BXH   4,6,KEYGONE                                              IONQ1800
         BXLE  9,10,MOVEKEY                                             IONQ1810
         LA    11,40(11) NEXT LINE                                      IONQ1820
         B     *-8 SKIP EOL                                             IONQ1830
*        RELATIVE FILE KEY HANDLER                                      IONQ1840
         CNOP  0,8                                                      IONQ1850
WORD     DS    CL8                                                      IONQ1860
CHAR     DS    CL16                                                     IONQ1870
FORTKEY  L     7,0(4) GET KEY VALUE                                     IONQ1880
         CVD   7,WORD                                                   IONQ1890
         UNPK  CHAR,WORD                                                IONQ1900
         OI    CHAR+15,X'F0' SIGN=ZONE                                  IONQ1910
         LA    4,CHAR RESET KEY POINTER                                 IONQ1920
         LA    7,16 SET LENGTH                                          IONQ1930
SUPP     CLI   0(4),X'F0' SUPPRESS LEADING ZEROES                       IONQ1940
```

```
         BNE SETKEY                                                     IONQ1950
         LA    4,1(4) INCR ADDR                                         IONQ1960
         BCTR  7,0 DECR LEN                                             IONQ1970
         LTR   7,7                                                      IONQ1980
         BP    SUPP IF NOT ZERO VALUE                                   IONQ1990
         LA    7,4                                                      IONQ2000
         LA    4,*+8 GET TEXT                                           IONQ2010
         B     SETKEY                                                   IONQ2020
         DC    C'ZERO'                                                  IONQ2030
         EJECT                                                          IONQ2040
*        SET   RETURN CHAR FOR 1053                                     IONQ2050
KEYGONE  LA    10,40  LINE INCR FOR EOL SYMBOL                          IONQ2060
         LR    3,2    SAVE SCPBF START ADDR                             IONQ2070
SETEOL   MVI   39(2),X'5C' *TO BE CHANGED TO X'0A' FOR 1053/2260        IONQ2080
         BXLE  2,10,SETEOL                                              IONQ2090
*        SET   OUTPUT LEN IN ECB                                        IONQ2100
         SR    2,3    GET MSG LEN                                       IONQ2110
         L     1,0(1) FIND 'REG8'                                       IONQ2120
         ST    2,8(1) SET SCOPE LEN                                     IONQ2130
* *END* *ERROR*PROCESSOR                                                IONQ2140
*      * GET   BACK                                                     IONQ2150
         EX    0,RESET13 TO POINT TO RIGHT SAVE AREA                    IONQ2160
         B     RTRN TO RETURN TO TRMINAL MONITOR                        IONQ2170
*      * END   *                                                        IONQ2180
         SPACE 5                                                        IONQ2190
         LTORG                                                          IONQ2200
         EJECT                                                          IONQ2210
         END                                                            IONQ2220
```

```
IOTA     TITLE 'LOCATOR LIST FOR I/O ROUTINES                W W JONES'
RTLIST   START 0
         DC    V(AUTSEQ)
         DC    V(TITSEQ)
         DC    V(SUBSEQ)
         DC    V(DSCSEQ)
         DC    V(ACNFOR)
         DC    V(AUTFOR)
         DC    V(TI1FOR)
         DC    V(TI2FOR)
         DC    V(SU1FOR)
         DC    V(SU2FOR)
         DC    V(RDVEN)
         DC    V(RDPLC)
         DC    V(RDPUB)
         DC    V(RDALP)
         DC    V(RDNUM)
         DC    V(RFILEF)
         DC    V(RDSCRT)
         DC    V(WTSCRT)
         DC    V(FLNMLST)  WHEN THE RELATIVE POSITION OF THIS ENTRY IS 1
               CHANGED, ALSO CHANGE ENTRY IN IOENQ CALLED FILELIST
         DC    V(SCCONBLK)  SCRATCH FILE CONTROL BLOCK
         DC    V(RDPLAC)
         DC    V(RDPUBL)
         DC    V(RAUTIO)
         DC    V(RTITIO)
         DC    V(RSUBIO)
         DC    V(RDSCIO)
         DC    V(RWACN1)
         DC    V(WTAUT)
         DC    V(WTSUB1)
         DC    V(WTSUB2)
         DC    V(WTTI1)
         DC    V(WTTI2)
         DC    V(WTACN)
         DC    V(RWACNU)
         DC    V(RWAUTU)
         DC    V(RWT1UP)
         DC    V(RWT2UP)
         DC    V(RWSB1U)
         DC    V(RWSB2U)
         END
```

```
IDENTIFICATION DIVISION.
PROGRAM-ID. 'OPNCLO'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-360.
OBJECT-COMPUTER. IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
77  TEST                    PICTURE 9.
LINKAGE SECTION.
PROCEDURE DIVISION.
    ENTER LINKAGE.
    ENTRY 'OPNRN'.
    ENTER COBOL.
    ENTER LINKAGE.
    CALL 'AUTOPN'.
    ENTER COBOL.
    ENTER LINKAGE.
    CALL 'TITOPN'.
    ENTER COBOL.
    ENTER LINKAGE.
    CALL 'SUBOPN'.
    ENTER COBOL.
    ENTER LINKAGE.
    CALL 'DSCOPN'.
    ENTER COBOL.
    ENTER LINKAGE.
    CALL 'OPEALP'.
    ENTER COBOL.
    ENTER LINKAGE.
    CALL 'OPENUM'.
    ENTER COBOL.
    ENTER LINKAGE.
    CALL 'OPNAUTIO'.
    ENTER COBOL.
    ENTER LINKAGE.
```

```
         CALL 'OPNTITIO'.
         ENTER COBOL.
         ENTER LINKAGE.
         CALL 'OPNSUBIO'.
         ENTER COBOL.
         ENTER LINKAGE.
         CALL 'OPNDSCIO'.
         ENTER COBOL.
     A1.
         ENTER LINKAGE.
         RETURN.
         ENTER COBOL.
         ENTER LINKAGE.
         ENTRY 'CLSRN'.
         ENTER COBOL.
         ENTER LINKAGE.
         CALL 'AUTCLO'.
         ENTER COBOL.
         ENTER LINKAGE.
         CALL 'TITCLO'.
         ENTER COBOL.
         ENTER LINKAGE.
         CALL 'SUBCLO'.
         ENTER COBOL.
         ENTER LINKAGE.
         CALL 'DSCCLO'.
         ENTER COBOL.
         ENTER LINKAGE.
         CALL 'CLOALP'.
         ENTER COBOL.
         ENTER LINKAGE.
         CALL 'CLONUM'.
         ENTER COBOL.
         ENTER LINKAGE.
         CALL 'CLOAUTIO'.
         ENTER COBOL.
```

```
ENTER LINKAGE.
CALL 'CLOTITIO'.
ENTER COBOL.
ENTER LINKAGE.
CALL 'CLOSUBIO'.
ENTER COBOL.
ENTER LINKAGE.
CALL 'CLODSCIO'.
ENTER COBOL.
GO TO A1.
```

```
IDENTIFICATION DIVISION.
PROGRAM-ID. 'SEQAUT'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-360.
OBJECT-COMPUTER. IBM-360.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT AUTH-IN ASSIGN TO 'SYS005' DIRECT-ACCESS 2311
        ORGANIZATION IS INDEXED ACCESS IS RANDOM
        RECORD KEY IS AUTH1
        SYMBOLIC KEY IS AUTH-SYM-KEY.
DATA DIVISION.
FILE SECTION.
FD  AUTH-IN
    LABEL RECORDS ARE STANDARD
    RECORDING MODE IS F
    DATA RECORD IS AUTH-REC.
01  AUTH-REC.
    02  FILLER              PICTURE X.
    02  AUTH1               PICTURE X(213).
    02  NDEX-FIRST-NAME     PICTURE 9(6).
WORKING-STORAGE SECTION.
77  AUTH-SYM-KEY            PICTURE X(213).
LINKAGE SECTION.
01  S-KEY.
    02  SYM-KEY             PICTURE X(213).
01  BUFER.
    02  ER                  PICTURE X.
    02  AUT                 PICTURE X(213).
    02  N-F-N               PICTURE 9(6).
PROCEDURE DIVISION.
    ENTER LINKAGE.
    ENTRY 'AUTSEQ' USING BUFER S-KEY.
    ENTER COBOL.
    MOVE S-KEY TO AUTH-SYM-KEY AUT.
```

```
          READ AUTH-IN INVALID KEY GO TO ERR.
          MOVE AUTH1 TO AUT.
          MOVE NDEX-FIRST-NAME TO N-F-N.
          MOVE 0 TO ER.
      A1.
          ENTER LINKAGE.
          RETURN.
          ENTER COBOL.
      ERR.
          MOVE '1' TO ER.
          GO TO A1.
          ENTER LINKAGE.
          ENTRY 'AUTOPN'.
          ENTER COBOL.
          OPEN INPUT AUTH-IN.
          GO TO A1.
          ENTER LINKAGE.
          ENTRY 'AUTCLO'.
          ENTER COBOL.
          CLOSE AUTH-IN.
          GO TO A1.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'AUTIO'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 INPUT-OUTPUT SECTION.
001080 FILE-CONTROL.
001090     SELECT AUTH-IN ASSIGN TO 'SYS009' DIRECT-ACCESS 2311
001100         ORGANIZATION IS INDEXED ACCESS IS RANDOM
001110         RECORD KEY IS AUTH1
001120         SYMBOLIC KEY IS AUTH-SYM-KEY.
001130 DATA DIVISION.
001140 FILE SECTION.
001150 FD  AUTH-IN
001160     LABEL RECORDS ARE STANDARD
001170     RECORDING MODE IS F
001180     DATA RECORD IS AUTH-REC.
001190 01  AUTH-REC.
001200     02  IND                    PICTURE X.
001210     02  AUTH1                  PICTURE X(213).
001220     02  NDEX-FIRST-NAME        PICTURE 9(6).
001230 WORKING-STORAGE SECTION.
001240 77  AUTH-SYM-KEY               PICTURE X(213).
001241 01  CNTRL-WRD.
001242     02  H-WRD1                 PICTURE X.
001243     02  H-WRD2                 PICTURE X.
001244 01  CONTRL-WRD REDEFINES CNTRL-WRD.
001245     02  WORD1                  PICTURE 9(3) COMPUTATIONAL.
001250 LINKAGE SECTION.
002010 01  S-KEY.
002020     02  SYM-KEY                PICTURE X(213).
002030 01  BUFER.
002040     02  ER                     PICTURE X.
002050     02  AUT                    PICTURE X(213).
002060     02  N-F-N                  PICTURE 9(6).
```

```
002070 PROCEDURE DIVISION.
002080     ENTER LINKAGE.
002090     ENTRY 'RAUTIO' USING BUFER S-KEY.
002100     ENTER COBOL.
002101     MOVE 189 TO WORD1.
002110     IF ER = '1' GO TO READ-IT.
002120     IF ER = '2' GO TO RE-WRITE.
002130     IF ER = '3' GO TO WRITE-IT.
002131     IF ER = HIGH-VALUE GO TO RE-WRITE.
002140 E.  MOVE '1' TO ER.
002150 E1. ENTER LINKAGE.
002160     RETURN.
002170     ENTER COBOL.
002180 READ-IT.
002190     MOVE S-KEY TO AUTH-SYM-KEY AUT.
002200     READ AUTH-IN INVALID KEY GO TO E.
002210     MOVE AUTH1 TO AUT.
002220     MOVE NDEX-FIRST-NAME TO N-F-N.
002230     MOVE H-WRD2 TO ER.
002240     GO TO E1.
002250 RE-WRITE.
003010     MOVE S-KEY TO AUTH-SYM-KEY.
003020     MOVE BUFER TO AUTH-REC.
003030     REWRITE AUTH-REC INVALID KEY GO TO E.
003040     MOVE '0' TO ER.
003050     GO TO E1.
003060 WRITE-IT.
003070     MOVE S-KEY TO AUTH-SYM-KEY.
003080     MOVE BUFER TO AUTH-REC.
003090     WRITE AUTH-REC INVALID KEY GO TO E.
003100     MOVE '0' TO ER
003110     GO TO E1.
003120     ENTER LINKAGE.
003130     ENTRY 'OPNAUTIO'.
003140     ENTER COBOL.
003150     OPEN I-O AUTH-IN.
```

```
003160     GO TO E1.
003170     ENTER LINKAGE.
003180     ENTRY 'CLOAUTIO'.
003190     ENTER COBOL.
003200     CLOSE AUTH-IN.
003210     GO TO E1.
```

```
IDENTIFICATION DIVISION.
PROGRAM-ID. 'SEQTIT'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-360.
OBJECT-COMPUTER. IBM-360.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT TITL-IN ASSIGN TO 'SYS006' DIRECT-ACCESS 2311
        ORGANIZATION IS INDEXED ACCESS IS RANDOM
        RECORD KEY IS TITL1
        SYMBOLIC KEY IS TITL-SYM-KEY.
DATA DIVISION.
FILE SECTION.
FD  TITL-IN
    LABEL RECORDS ARE STANDARD
    RECORDING MODE IS F
    DATA RECORD IS TITL-REC.
01  TITL-REC.
    02  FILLER                PICTURE X.
    02  TITL1                 PICTURE X(240).
    02  NDEX-FIRST-NAME       PICTURE 9(6).
WORKING-STORAGE SECTION.
77  TITL-SYM-KEY              PICTURE X(240).
LINKAGE SECTION.
01  S-KEY.
    02  SYM-KEY               PICTURE X(240).
01  BUFER.
    02  ER                    PICTURE X.
    02  TIL                   PICTURE X(240).
    02  N-F-N                 PICTURE 9(6).
PROCEDURE DIVISION.
    ENTER LINKAGE.
    ENTRY 'TITSEQ' USING BUFER S-KEY.
    ENTER COBOL.
    MOVE S-KEY TO TITL-SYM-KEY TIL.
```

```
          READ TITL-IN INVALID KEY GO TO ERR.
          MOVE TITL1 TO TIL.
          MOVE NDEX-FIRST-NAME TO N-F-N.
          MOVE 0 TO ER.
      A1.
          ENTER LINKAGE.
          RETURN.
          ENTER COBOL.
      ERR.
          MOVE '1' TO ER.
          GO TO A1.
          ENTER LINKAGE.
          ENTRY 'TITOPN'.
          ENTER COBOL.
          OPEN INPUT TITL-IN.
          GO TO A1.
          ENTER LINKAGE.
          ENTRY 'TITCLO'.
          ENTER COBOL.
          CLOSE TITL-IN.
          GO TO A1.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'TITLIO'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 INPUT-OUTPUT SECTION.
001080 FILE-CONTROL.
001090     SELECT TITL-IN ASSIGN TO 'SYS010' DIRECT-ACCESS 2311
001100         ORGANIZATION IS INDEXED ACCESS IS RANDOM
001110         RECORD KEY IS TITL1
001120         SYMBOLIC KEY IS TITL-SYM-KEY.
001130 DATA DIVISION.
001140 FILE SECTION.
001150 FD  TITL-IN
001160     LABEL RECORDS ARE STANDARD
001170     RECORDING MODE IS F
001180     DATA RECORD IS TITL-REC.
001190 01  TITL-REC.
001200     02  IND                    PICTURE X.
001210     02  TITL1                  PICTURE X(240).
001220     02  NDEX-FIRST-NAME        PICTURE 9(6).
001230 WORKING-STORAGE SECTION.
001240 77  TITL-SYM-KEY               PICTURE X(240).
001250 01  CNTRL-WRD.
001251     02  H-WRD1                 PICTURE X.
001252     02  H-WRD2                 PICTURE X.
001253 01  CONTRL-WRD REDEFINES CNTRL-WRD.
001254     02  WORD1                  PICTURE 9(3) COMPUTATIONAL.
002010 LINKAGE SECTION.
002020 01  S-KEY.
002030     02  SYM-KEY                PICTURE X(240).
002040 01  BUFER.
002050     02  ER                     PICTURE X.
002060     02  TIL                    PICTURE X(240).
002070     02  N-F-N                  PICTURE 9(6).
```

```
002080 PROCEDURE DIVISION.
002090     ENTER LINKAGE.
002100     ENTRY 'RTITIO' USING BUFER S-KEY.
002110     ENTER COBOL.
002111     MOVE 189 TO WORD1.
002120     IF ER = '1' GO TO READ-IT.
002130     IF ER = '2' GO TO RE-WRITE.
002140     IF ER = '3' GO TO WRITE-IT.
002141     IF ER = HIGH-VALUE GO TO RE-WRITE.
002150 E.  MOVE '1' TO ER.
002160 E1. ENTER LINKAGE.
002170     RETURN.
002180     ENTER COBOL.
002190 READ-IT.
002200     MOVE S-KEY TO TITL-SYM-KEY TIL.
002210     READ TITL-IN INVALID KEY GO TO E.
002220     MOVE TITL1 TO TIL.
002230     MOVE NDEX-FIRST-NAME TO N-F-N.
002240     MOVE H-WRD2 TO ER.
002250     GO TO E1.
003010 RE-WRITE.
003020     MOVE S-KEY TO TITL-SYM-KEY.
003030     MOVE BUFER TO TITL-REC.
003040     REWRITE TITL-REC INVALID KEY GO TO E.
003050     MOVE '0' TO ER.
003060     GO TO E1.
003070 WRITE-IT.
003080     MOVE S-KEY TO TITL-SYM-KEY.
003090     MOVE BUFER TO TITL-REC.
003100     WRITE TITL-REC INVALID KEY GO TO E.
003110     MOVE '0' TO ER.
003120     GO TO E1.
003130     ENTER LINKAGE.
003140     ENTRY 'OPNTITIO'.
003150     ENTER COBOL.
003160     OPEN I-O TITL-IN.
```

```
003170     GO TO E1.
003180     ENTER LINKAGE.
003190     ENTRY 'CLOTITIO'.
003200     ENTER COBOL.
003210     CLOSE TITL-IN.
003220     GO TO E1.
```

```
IDENTIFICATION DIVISION.
PROGRAM-ID. 'SEQSUB'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-360.
OBJECT-COMPUTER. IBM-360.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT SUBJ-IN ASSIGN TO 'SYS007' DIRECT-ACCESS 2311
        ORGANIZATION IS INDEXED ACCESS IS RANDOM
        RECORD KEY IS SUBJ1
        SYMBOLIC KEY IS SUBJ-SYM-KEY.
DATA DIVISION.
FILE SECTION.
FD  SUBJ-IN
    LABEL RECORDS ARE STANDARD
    RECORDING MODE IS F
    DATA RECORD IS SUBJ-REC.
01  SUBJ-REC.
    02  FILLER                  PICTURE X.
    02  SUBJ1                   PICTURE X(80).
    02  NDEX-FIRST-NAME         PICTURE 9(6).
WORKING-STORAGE SECTION.
77  SUBJ-SYM-KEY                PICTURE X(80).
LINKAGE SECTION.
01  S-KEY.
    02  SYM-KEY                 PICTURE X(80).
01  BUFER.
    02  ER                      PICTURE X.
    02  SUB                     PICTURE X(80).
    02  N-F-N                   PICTURE 9(6).
PROCEDURE DIVISION.
    ENTER LINKAGE.
    ENTRY 'SUBSEQ' USING BUFER S-KEY.
    ENTER COBOL.
    MOVE S-KEY TO SUBJ-SYM-KEY SUB.
```

```
          READ SUBJ-IN INVALID KEY GO TO ERR.
          MOVE SUBJ1 TO SUB.
          MOVE NDEX-FIRST-NAME TO N-F-N.
          MOVE 0 TO ER.
      A1.
          ENTER LINKAGE.
          RETURN.
          ENTER COBOL.
      ERR.
          MOVE '1' TO ER.
          GO TO A1.
          ENTER LINKAGE.
          ENTRY 'SUBOPN'.
          ENTER COBOL.
          OPEN INPUT SUBJ-IN.
          GO TO A1.
          ENTER LINKAGE.
          ENTRY 'SUBCLO'.
          ENTER COBOL.
          CLOSE SUBJ-IN.
          GO TO A1.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'SUBJIO'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 INPUT-OUTPUT SECTION.
001080 FILE-CONTROL.
001090     SELECT SUBJ-IN ASSIGN TO 'SYS011' DIRECT-ACCESS 2311
001100         ORGANIZATION IS INDEXED ACCESS IS RANDOM
001110         RECORD KEY IS SUBJ1
001120         SYMBOLIC KEY IS SUBJ-SYM-KEY.
001130 DATA DIVISION.
001140 FILE SECTION.
001150 FD  SUBJ-IN
001160     LABEL RECORDS ARE STANDARD
001170     RECORDING MODE IS F
001180     DATA RECORD IS SUBJ-REC.
001190 01  SUBJ-REC.
001200     02  IND                     PICTURE X.
001210     02  SUBJ1                   PICTURE X(80).
001220     02  NDEX-FIRST-NAME         PICTURE 9(6).
001230 WORKING-STORAGE SECTION.
001240 77  SUBJ-SYM-KEY                PICTURE X(80).
001250 01  CNTRL-WRD.
001251     02  H-WRD1                  PICTURE X.
001252     02  H-WRD2                  PICTURE X.
001253 01  CONTRL-WRD REDEFINES CNTRL-WRD.
001254     02  WORD1                   PICTURE 9(3) COMPUTATIONAL.
002010 LINKAGE SECTION.
002020 01  S-KEY.
002030     02  SYM-KEY                 PICTURE X(80).
002040 01  BUFER.
002050     02  ER                      PICTURE X.
002060     02  SUB                     PICTURE X(80).
002070     02  N-F-N                   PICTURE 9(6).
```

```
002080 PROCEDURE DIVISION.
002090     ENTER LINKAGE.
002100     ENTRY 'RSUBIO' USING BUFER S-KEY.
002110     ENTER COBOL.
002111     MOVE 189 TO WORD1.
002120     IF ER = '1' GO TO READ-IT.
002130     IF ER = '2' GO TO RE-WRITE.
002140     IF ER = '3' GO TO WRITE-IT.
002141     IF ER = HIGH-VALUE GO TO RE-WRITE.
002150 E.  MOVE '1' TO ER.
002160 E1. ENTER LINKAGE.
002170     RETURN.
002180     ENTER COBOL.
002190 READ-IT.
002200     MOVE S-KEY TO SUBJ-SYM-KEY SUB.
002210     READ SUBJ-IN INVALID KEY GO TO E.
002220     MOVE SUBJ1 TO SUB.
002230     MOVE NDEX-FIRST-NAME TO N-F-N.
002240     MOVE H-WRD2 TO ER.
002250     GO TO E1.
003010 RE-WRITE.
003020     MOVE S-KEY TO SUBJ-SYM-KEY.
003030     MOVE BUFER TO SUBJ-REC.
003040     REWRITE SUBJ-REC INVALID KEY GO TO E.
003050     MOVE '0' TO ER.
003060     GO TO E1.
003070 WRITE-IT.
003080     MOVE S-KEY TO SUBJ-SYM-KEY.
003090     MOVE BUFER TO SUBJ-REC.
003100     WRITE SUBJ-REC INVALID KEY GO TO E.
003110     MOVE '0' TO ER.
003120     GO TO E1.
003130     ENTER LINKAGE.
003140     ENTRY 'OPNSUBIO'.
003150     ENTER COBOL.
003160     OPEN I-O SUBJ-IN.
```

```
003170      GO TO E1.
003180      ENTER LINKAGE.
003190      ENTRY 'CLOSUBIO'.
003200      ENTER COBOL.
003210      CLOSE SUBJ-IN.
003220      GO TO E1.
```

```
IDENTIFICATION DIVISION.
PROGRAM-ID. 'SEQDSC'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-360.
OBJECT-COMPUTER. IBM-360.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT DESC-FILE ASSIGN TO 'SYS008' DIRECT-ACCESS 2311
        ORGANIZATION IS INDEXED ACCESS IS RANDOM
        RECORD KEY IS DESC-SEEK-KEY
        SYMBOLIC KEY IS DESC-SYM-KEY.
DATA DIVISION.
FILE SECTION.
FD  DESC-FILE
    LABEL RECORDS ARE STANDARD
    RECORDING MODE IS F
DATA RECORD IS DESC-REC.
01  DESC-REC.
    02  FILLER                PICTURE X.
    02  DESC-SEEK-KEY         PICTURE X(16).
    02  MIDL                  PICTURE X(412).
    02  NINS                  PICTURE 9(18).
    02  LSTS                  PICTURE X(43).
WORKING-STORAGE SECTION.
77  DESC-SYM-KEY              PICTURE X(16).
LINKAGE SECTION.
01  S-KEY.
    02  SYM-KEY               PICTURE X(16).
01  BUFER.
    02  ER                    PICTURE X.
    02  DESC                  PICTURE X(16).
    02  MID                   PICTURE X(412).
    02  NS                    PICTURE 9(18).
    02  LST                   PICTURE X(43).
PROCEDURE DIVISION.
```

```
     ENTER LINKAGE.
     ENTRY 'DSCSEQ' USING BUFER S-KEY.
     ENTER COBOL.
     MOVE S-KEY TO DESC-SYM-KEY DESC-SEEK-KEY.
     READ DESC-FILE INVALID KEY GO TO ERR.
     MOVE DESC-SEEK-KEY TO DESC. MOVE MIDL TO MID.
     MOVE NINS TO NS. MOVE LSTS TO LST.
     MOVE 0 TO ER.
A1.
     ENTER LINKAGE.
     RETURN.
     ENTER COBOL.
ERR.
     MOVE '1' TO ER.
     GO TO A1.
     ENTER LINKAGE.
     ENTRY 'DSCOPN'.
     ENTER COBOL.
     OPEN INPUT DESC-FILE.
     GO TO A1.
     ENTER LINKAGE.
     ENTRY 'DSCCLO'.
     ENTER COBOL.
     CLOSE DESC-FILE.
     GO TO A1.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'DESCIO'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 INPUT-OUTPUT SECTION.
001080 FILE-CONTROL.
001090     SELECT DESC-FILE ASSIGN TO 'SYS012' DIRECT-ACCESS 2311
001100     ORGANIZATION IS INDEXED ACCESS IS RANDOM
001110     RECORD KEY IS DESC-SEEK-KEY
001120     SYMBOLIC KEY IS DESC-SYM-KEY.
001130 DATA DIVISION.
001140 FILE SECTION.
001150 FD  DESC-FILE
001160     LABEL RECORDS ARE STANDARD
001170     RECORDING MODE IS F
001180     DATA RECORD IS DESC-REC.
001190 01  DESC-REC.
001200     02  IND                 PICTURE X.
001210     02  DESC-SEEK-KEY       PICTURE X(16).
001220     02  MIDL                PICTURE X(412).
001230     02  NINS                PICTURE 9(18).
001240     02  LSTS                PICTURE X(43).
001250 WORKING-STORAGE SECTION.
002010 77  DESC-SYM-KEY            PICTURE X(16).
002020 01  CNTRL-WRD.
002021     02  H-WRD1              PICTURE X.
002022     02  H-WRD2              PICTURE X.
002023 01  CONTRL-WRD REDEFINES CNTRL-WRD.
002024     02  WORD1               PICTURE 9(3) COMPUTATIONAL.
002030 LINKAGE SECTION.
002040 01  S-KEY.
002050     02  SYM-KEY             PICTURE X(16).
002060 01  BUFER.
002070     02  ER                  PICTURE X.
```

```
002080     02  DESC                   PICTURE X(16).
002090     02  MID                    PICTURE X(412).
002100     02  NS                     PICTURE 9(18).
002110     02  LST                    PICTURE X(43).
002120 PROCEDURE DIVISION.
002130     ENTER LINKAGE.
002140     ENTRY 'RDSCIO' USING BUFER S-KEY.
002150     ENTER COBOL.
002151     MOVE 189 TO WORD1.
002160     IF ER = '1' GO TO READ-IT.
002170     IF ER = '2' GO TO RE-WRITE.
002180     IF ER = '3' GO TO WRITE-IT.
002181     IF ER = HIGH-VALUE GO TO RE-WRITE.
002190 E.  MOVE '1' TO ER.
002200 E1. ENTER LINKAGE.
002210     RETURN.
002220     ENTER COBOL.
002230 READ-IT.
002240     MOVE S-KEY TO DESC-SYM-KEY DESC-SEEK-KEY.
002250     READ DESC-FILE INVALID KEY GO TO E.
003010     MOVE DESC-REC TO BUFER.
003020     MOVE H-WRD2 TO ER.
003030     GO TO E1.
003040 RE-WRITE.
003050     MOVE S-KEY TO DESC-SYM-KEY.
003051     IF ER NOT = HIGH-VALUE MOVE SPACES TO ER.
003060     MOVE BUFER TO DESC-REC.
003070     REWRITE DESC-REC INVALID KEY GO TO E.
003080     MOVE '0' TO ER.
003090     GO TO E1.
003100 WRITE-IT.
003110     MOVE S-KEY TO DESC-SYM-KEY.
003111     MOVE SPACES TO ER.
003120     MOVE BUFER TO DESC-REC.
003130     WRITE DESC-REC INVALID KEY GO TO E.
003140     MOVE '0' TO ER.
```

```
003150      GO TO E1.
003160      ENTER LINKAGE.
003170      ENTRY 'OPNDSCIO'.
003180      ENTER COBOL.
003190      OPEN I-O DESC-FILE
003200      GO TO E1.
003210      ENTER LINKAGE.
003220      ENTRY 'CLODSCIO'.
003230      ENTER COBOL.
003240      CLOSE DESC-FILE.
003250      GO TO E1.
```

```
      SUBROUTINE RWAUTU(BUF,KEY)
      DIMENSION IHALF(2),BUF(131),IBUF(60)
      INTEGER*2 IHALF
      INTEGER BUF
      EQUIVALENCE (IBUF(1),IHALF(1))
      DEFINE FILE 4(999,236,E,KEY)
      DATA ISWIT/1/
      IF(ISWIT)1,2,1
    1 READ(4'KEY,100) (IBUF(I),I=2,60)
      IHALF(1)=-16640
      IHALF(2)=236
      ISWIT=0
      DO 3 I=1,60
    3 BUF(I)=IBUF(I+1)
C   3 BUF(I)=IBUF(I)
  100 FORMAT(53A4,1A1,3I5,2A4)
      RETURN
    2 WRITE(4'KEY,100) (BUF(I),I=1,59)
      ISWIT=1
      RETURN
      END
      SUBROUTINE RWSB1U(BUF,KEY)
      DIMENSION IHALF(2),BUF(131),IBUF(25)
      INTEGER*2 IHALF
      INTEGER BUF
      EQUIVALENCE (IBUF(1),IHALF(1))
      DEFINE FILE 13(999,96,E,KEY)
      DATA ISWIT/1/
      IF(ISWIT)1,2,1
    1 READ(13'KEY,100) (IBUF(I),I=2,25)
      IHALF(1)=-16640
      IHALF(2)=96
      ISWIT=0
      DO 3 I=1,25
    3 BUF(I)=IBUF(I+1)
C   3 BUF(I)=IBUF(I)
```

4

```
  100 FORMAT(20A4,I6,2A4,A2)
      RETURN
    2 WRITE(13'KEY,100) (BUF(I),I=1,24)
      ISWIT=1
      RETURN
      END
      SUBROUTINE RWSB2U(BUF,KEY)
      DIMENSION IHALF(2),BUF(131),IBUF(5)
      INTEGER*2 IHALF
      INTEGER BUF
      EQUIVALENCE (IBUF(1),IHALF(1))
      DEFINE FILE 14(999,24,E,KEY)
      DATA ISWIT/1/
      IF(ISWIT)1,2,1
    1 READ(14'KEY,100) (IBUF(I),I=2,5)
      IHALF(1)=-16640
      IHALF(2)=24
      ISWIT=0
      DO 3 I=1,5
    3 BUF(I)=IBUF(I+1)
C   3 BUF(I)=IBUF(I)
  100 FORMAT(4I6)
      RETURN
    2 WRITE(14'KEY,100) (BUF(I),I=1,4)
      ISWIT=1
      RETURN
      END
      SUBROUTINE RWT1UP(BUF,KEY)
      DIMENSION IHALF(2),BUF(131),IBUF(62)
      INTEGER*2 IHALF
      INTEGER BUF
      EQUIVALENCE (IBUF(1),IHALF(1))
      DEFINE FILE 3(999,246,E,KEY)
      DATA ISWIT/1/
      IF(ISWIT)1,2,1
    1 READ(3'KEY,100) (IBUF(I),I=2,62)
```

```
      IHALF(1)=-16640
      IHALF(2)=246
      ISWIT=0
      DO 3 I=1,62
    3 BUF(I)=IBUF(I+1)
C   3 BUF(I)=IBUF(I)
  100 FORMAT(60A4,I6)
      RETURN
    2 WRITE(3'KEY,100) (BUF(I),I=1,61)
      ISWIT=1
      RETURN
      END
      SUBROUTINE RWT2UP(BUF,KEY)
      DIMENSION IHALF(2),BUF(131),IBUF(4)
      INTEGER*2 IHALF
      INTEGER BUF
      EQUIVALENCE (IBUF(1),IHALF(1))
      DEFINE FILE 5(999,18,E,KEY)
      DATA ISWIT/1/
      IF(ISWIT)1,2,1
    1 READ(5'KEY,100) (IBUF(I),I=2,4)
      IHALF(1)=-16640
      IHALF(2)=18
      ISWIT=0
      DO 3 I=1,4
    3 BUF(I)=IBUF(I+1)
C   3 BUF(I)=IBUF(I)
  100 FORMAT(3I6)
      RETURN
    2 WRITE(5'KEY,100) (BUF(I),I=1,3)
      ISWIT=1
      RETURN
      END
      SUBROUTINE RWACNU(BUF,KEY)
      DIMENSION IHALF(2),BUF(131),IBUF(11)
      INTEGER*2 IHALF
      INTEGER BUF
      EQUIVALENCE (IBUF(1),IHALF(1))
      DEFINE FILE 2(999,52,E,KEY)
      DATA ISWIT/1/
      IF(ISWIT)1,2,1
    1 READ(2'KEY,100) (IBUF(I),I=2,11)
      IHALF(1)=-16640
      IHALF(2)=52
      ISWIT=0
      DO 3 I=1,11
    3 BUF(I)=IBUF(I+1)
C   3 BUF(I)=IBUF(I)
  100 FORMAT(4A4,6I6)
      RETURN
    2 WRITE(2'KEY,100) (BUF(I),I=1,10)
      ISWIT=1
      RETURN
      END
      SUBROUTINE WTAUT(BUF,KEY)
      DIMENSION BUF(59)
      INTEGER BUF
      DEFINE FILE 4(999,236,E,KEY)
      WRITE(4'KEY,100) (BUF(I),I=1,59)
  100 FORMAT(53A4,1A1,3I5,2A4)
      RETURN
      END
      SUBROUTINE WTSUB1(BUF,KEY)
      INTEGER BUF
      DIMENSION BUF(24)
      DEFINE FILE 13(999,96,E,KEY)
      WRITE(13'KEY,100) (BUF(I),I=1,24)
  100 FORMAT(20A4,I6,2A4,A2)
      RETURN
      END
      SUBROUTINE WTSUB2(BUF,KEY)
      INTEGER BUF
      DIMENSION BUF(4)
      DEFINE FILE 14(999,24,E,KEY)
      WRITE(14'KEY,100) (BUF(I),I=1,4)
  100 FORMAT(4I6)
      RETURN
      END
      SUBROUTINE WTTI1(BUF,KEY)
      INTEGER BUF
      DIMENSION BUF(61)
      DEFINE FILE 3(999,246,E,KEY)
      WRITE(3'KEY,100) (BUF(I),I=1,61)
  100 FORMAT(60A4,I6)
      RETURN
      END
      SUBROUTINE WTTI2(BUF,KEY)
      INTEGER BUF
      DIMENSION BUF(3)
      DEFINE FILE 5(999,18,E,KEY)
      WRITE(5'KEY,100) (BUF(I),I=1,3)
  100 FORMAT(3I6)
      RETURN
      END
      SUBROUTINE WTACN(BUF,KEY)
      INTEGER BUF
      DIMENSION BUF(10)
      DEFINE FILE 2(999,52,E,KEY)
      WRITE(2'KEY,100) (BUF(I),I=1,10)
  100 FORMAT(4A4,6I6)
      RETURN
      END
      SUBROUTINE RDPLAC(N,NPLACE)
      DIMENSION NPLACE(30),MPLACE(30)
      DATA NSPAC/'    '/
      DEFINE FILE 7(99,120,E,NOREC)
      NOREC=1
   30 READ(7'NOREC,1) (MPLACE(I),I=1,30)
```

```
 1 FORMAT(30A4)
   DO 10 I=2,30
   IF(MPLACE(I)-NSPAC)11,10,11
10 CONTINUE
   NOREC=NOREC-1
   GO TO 20
11 DO 12 I=1,30
   IF(NPLACE(I)-MPLACE(I))30,12,30
12 CONTINUE
14 N=NOREC-1
   RETURN
20 WRITE(7'NOREC,1) (NPLACE(I),I=1,30)
   GO TO 14
   END
   SUBROUTINE RDPUBL(N,NPUB)
   DIMENSION NPUB(17),MPUB(17)
   DATA NSPAC/'    '/
   DEFINE FILE 6(200,68,E,NOREC)
   NOREC=1
30 READ(6'NOREC,1) (MPUB(I),I=1,17)
 1 FORMAT(17A4)
   DO 10 I=2,17
   IF(MPUB(I)-NSPAC)11,10,11
10 CONTINUE
   NOREC=NOREC-1
   GO TO 20
11 DO 12 I=1,17
   IF(NPUB(I)-MPUB(I))30,12,30
12 CONTINUE
   N=NOREC-1
   RETURN
20 WRITE(6'NOREC,1) (NPUB(I),I=1,17)
   GO TO 12
   END
```

```
      SUBROUTINE ACNFOR(BUF,KEY)
C     ACCN NUMBER = BUF(1)-BUF(4)
C     NEXT TITL1  = BUF(5)
C     NEXT SUBJ1  = BUF(6)
C     NEXT AUTHOR = BUF(7)
C     OTHER AUTH  = BUF(8)
C     AUTH LEVEL  = BUF(9)
C     NEXT ACCN   = BUF(10)
      INTEGER BUF
      DIMENSION BUF(123)
      DEFINE FILE 2(999,52,E,KEY)
    1 READ(2'KEY,100) (BUF(I),I=1,4),BUF(5),BUF(6),BUF(7),BUF(8),
     1BUF(9),BUF(10)
  100 FORMAT(4A4,6I6)
      RETURN
      END
      SUBROUTINE AUTFOR(BUF,KEY)
C     AUTHOR      =BUF(1)-BUF(54)
C     NEXT-AUTHOR =BUF(55)
C     ACCESSION A = BUF(56)
C     SEE INDEX   =BUF(57)
C     YEAR OF BRTH=BUF(58)-BUF(59)
      INTEGER BUF
      DIMENSION BUF(123)
      DEFINE FILE 4(999,236,E,KEY)
    1 READ(4'KEY,100) (BUF(I),I=1,54),BUF(55),BUF(56),BUF(57),
     1(BUF(J),J=58,59)
  100 FORMAT(53A4,1A1,3I5,2A4)
      RETURN
      END
      SUBROUTINE TI1FOR(BUF,KEY)
C     TITLE       =BUF(1)-BUF(60)
C     FIRST TITL2 = BUF(61)
      INTEGER BUF
      DIMENSION BUF(123)
      DEFINE FILE 3(999,246,E,KEY)
    1 READ(3'KEY,100) (BUF(I),I=1,60),BUF(61)
  100 FORMAT(60A4,I6)
      RETURN
      END
      SUBROUTINE TI2FOR(BUF,KEY)
C     NDEX TO ACCN= BUF(1)
C     NDEX TO NREC= BUF(2)
C     NDEX OTHER T= BUF(3)
      INTEGER BUF
      DIMENSION BUF(123)
      DEFINE FILE 5(999,18,E,KEY)
    1 READ(5'KEY,100) BUF(1),BUF(2),BUF(3)
  100 FORMAT(3I6)
      RETURN
      END
      SUBROUTINE SU1FOR(BUF,KEY)
C     SUBJECT     = BUF(1)-BUF(20)
C     NDEX TO SUB = BUF(21)
C     MESH        =BUF(22)-BUF(24)
      INTEGER BUF
      DIMENSION BUF(123)
      DEFINE FILE 13(999,96,E,KEY)
    1 READ(13'KEY,100) (BUF(I),I=1,20),BUF(21),(BUF(J),J=22,24)
  100 FORMAT(20A4,I6,2A4,A2)
      RETURN
      END
      SUBROUTINE WTSUB1(BUF,KEY)
C     SUBJECT     =BUF(1)-BUF(20)
C     NDEX TO SUB =BUF(21)
C     MESH        =BUF(22)-BUF(24)
      INTEGER BUF
      DIMENSION BUF(123)
      DEFINE FILE 13(999,96,E,KEY)
      WRITE(13'KEY,100) (BUF(I),I=1,20),BUF(21),(BUF(J),J=22,24)
  100 FORMAT(20A4,I6,2A4,A2)
      RETURN
      END
      SUBROUTINE WTSUB2(BUF,KEY)
C     NDEX TO ACN = BUF(1)
C     NDEX NEXT R2= BUF(2)
C     NDEX OTHER S= BUF(3)
C     PAGE        = BUF(4)
      INTEGER BUF
      DIMENSION BUF(123)
      DEFINE FILE 14(999,24,E,KEY)
      WRITE(14'KEY,100) BUF(1),BUF(2),BUF(3),BUF(4)
  100 FORMAT(4I6)
      RETURN
      END
      SUBROUTINE WTACN(BUF,KEY)
C     ACCN NUMBER = BUF(1)-BUF(4)
C     NDEX TITL1  = BUF(5)
C     NDEX SUBJ1  = BUF(6)
C     NDEX AUTHOR = BUF(7)
C     OTHER AUTH  = BUF(8)
C     AUTH LEVEL  = BUF(9)
C     NEXT ACCN   = BUF(10)
      INTEGER BUF
      DIMENSION BUF(123)
      DEFINE FILE 2(999,52,E,KEY)
      WRITE(2'KEY,100) (BUF(I),I=1,4),BUF(5),BUF(6),BUF(7),BUF(8),
     1BUF(9),BUF(10)
  100 FORMAT(4A4,6I6)
      RETURN
      END
      SUBROUTINE SU2FOR(BUF,KEY)
C     NDEX TO ACN = BUF(1)
C     NDEX NEXT R2= BUF(2)
C     NDEX OTHER S= BUF(3)
C     PAGE        = BUF(4)
      INTEGER BUF
      DIMENSION BUF(123)
      DEFINE FILE 14(999,24,E,KEY)
    1 READ(14'KEY,100) BUF(1),BUF(2),BUF(3),BUF(4)
  100 FORMAT(4I6)
      RETURN
      END
```

```
FLNM      TITLE 'FILE NAME LIST                    W W JONES'
FLNMLST   CSECT
          DS    0F
          DC    C'AUTORS '
          DC    X'D5'
          DC    C'TITLES '
          DC    X'F0'
          DC    C'SUBJFL '
          DC    X'50'
          DC    C'DESCFL '
          DC    X'10'
          DC    C'ACCN   '
          DC    X'00'
          DC    C'AUTGIN '
          DC    X'00'
          DC    C'TITL1  '
          DC    X'00'
          DC    C'TITL2  '
          DC    X'00'
          DC    C'SUBJ1  '
          DC    X'00'
          DC    C'SUBJ2  '
          DC    X'00'
          DC    C'VENFIL '
          DC    X'00'
          DC    C'LOCAT  '
          DC    X'00'
          DC    C'PUBLR  '
          DC    X'00'
          DC    C'DISKAL '
          DC    X'43'
          DC    C'DISKNM '
          DC    X'0D'
          DC    C'FILEF  '
          DC    X'00'
          DC    C'SCRATP '
DC     X'00'
DC     C'SCRATP  '
DC     X'00'
DC     C'DUMMY   '
DC     X'00'
END
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'M11'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001081 77  ACCN-HOLD                PICTURE 9(6) COMPUTATIONAL.
001082 77  SUB2                     PICTURE 9(5).
001090 77  SUB                      PICTURE 9(3) VALUE 1.
001100 77  RLDARN                   PICTURE 9(6) COMPUTATIONAL.
001110 77  FORT-KEY                 PICTURE 9(6) COMPUTATIONAL.
001120 77  DESC-SYM-KEY             PICTURE X(16) VALUE SPACES.
001130 77  SUB1                     PICTURE 9(3) VALUE 1.
001131 77  WHICH-ONE                PICTURE 9(6) COMPUTATIONAL.
001140 01  L-DATE.
001150         02  LOW-DATE         PICTURE X OCCURS 4 TIMES.
001160 01  H-DATE.
001170         02  HIGH-DATE        PICTURE X OCCURS 4 TIMES.
001171 01  POINTER2.
001172         02  PTARAY2 PICTURE 9(6) COMPUTATIONAL OCCURS 114 TIMES.
001172 01  LINK-OUT.
001173         02  BLK-LEN-OUT      PICTURE 9(4) COMPUTATIONAL.
001174         02  LNK-NO-OUT       PICTURE 9(4) COMPUTATIONAL.
002010 LINKAGE SECTION.
002020 77  RTLIST                   PICTURE 9(6) COMPUTATIONAL.
002030 77  SWIT                     PICTURE 9(8) COMPUTATIONAL.
002040 01  REG8.
002050         02  FILLER           PICTURE X(4).
002060         02  RORW             PICTURE 9(6) COMPUTATIONAL.
002070         02  LENG             PICTURE 9(6) COMPUTATIONAL.
002080 01  MESAGE.
002090         02  ID-NO            PICTURE X(10).
002100         02  FILLER           PICTURE X(2).
002110         02  COUNT-SAVE       PICTURE 9(6) COMPUTATIONAL.
```

```
002120     02  ALSO-COUNT          PICTURE 9(6) COMPUTATIONAL.
002130     02  POINTER1.
002131         03  PTARAY1 PICTURE 9(6) COMPUTATIONAL OCCURS 114 TIMES.
002132     02  LINK.
002133         03  BLK-LEN         PICTURE 9(4) COMPUTATIONAL.
002134         03  LNK-NO          PICTURE 9(4) COMPUTATIONAL.
002150 01  SCOPE.
002160     02  MES-LENG            PICTURE 9(5) COMPUTATIONAL.
002170     02  ANS.
002180         03  REPLY           PICTURE X OCCURS 40 TIMES.
002190     02  FILLER              PICTURE X(440).
002200 01  SCOPE-OUT REDEFINES SCOPE.
002201     02  OUT-LENG            PICTURE 9(5) COMPUTATIONAL.
002210     02 TOTAL-OUT            PICTURE X(480).
003010 01  DIRA.
003020     02  ACCES-NO            PICTURE X(16).
003030     02  NDEX-T-TITL         PICTURE 9(6) COMPUTATIONAL.
003040     02  NDEX-T-SUBJ         PICTURE 9(6) COMPUTATIONAL.
003050     02  NDEX-T-AUT          PICTURE 9(6) COMPUTATIONAL.
003060     02  NDEX-T-O-AUT        PICTURE 9(6) COMPUTATIONAL.
003070     02  A-LEV               PICTURE 9(6) COMPUTATIONAL.
003080     02  NDEX-T-N-ACCN       PICTURE 9(6) COMPUTATIONAL.
003090     02  FILLER              PICTURE X(450).
003100 01  DIRA1 REDEFINES DIRA.
003110     02  IOERR               PICTURE X.
003120     02  DESC-SEEK-KEY       PICTURE X(16).
003130     02  FILLER              PICTURE X(28).
003140     02  DATE-PUB1           PICTURE X(4).
003141     02  DATE-PUB2           PICTURE X(4).
003150     02  FILLER              PICTURE X(40).
003160     02  LANG                PICTURE X.
003170     02  FILLER              PICTURE X(396).
004010 PROCEDURE DIVISION.
004020     ENTER LINKAGE.
004030     ENTRY 'MOD11' USING REG8 RTLIST MESAGE SCOPE DIRA SWIT.
004040     ENTER COBOL.
```

```
004041      MOVE 0 TO ALSO-COUNT.
004050      IF REPLY (SUB) = 'N' GO TO TEST-NO.
004060      IF REPLY (SUB) = 'B' GO TO TEST-BOOK.
004070      IF REPLY (SUB) = 'J' GO TO TEST-JOUR.
004080      IF REPLY (SUB) = 'D' GO TO TEST-DATE.
004090      IF REPLY (SUB) = 'L' GO TO TEST-LANG.
004100 ER.  MOVE 'YOUR RESPONSE IS INCORRECT-TRY AGAIN.' TO TOTAL-OUT.
004101      MOVE 40 TO WHICH-ONE.
004102      ENTER LINKAGE.
004103      CALL 'GOSYM' USING TOTAL-OUT WHICH-ONE.
004104      ENTER COBOL.
004110      MOVE 8 TO RORW.
004120      MOVE 40 TO LENG.
004130      MOVE 33554442 TO SWIT.
004150 RTURN.
004160      ENTER LINKAGE.
004170      RETURN.
004180      ENTER COBOL.
004190 TEST-NO.
004200      ADD 1 TO SUB.
004210      IF REPLY (SUB) IS NOT EQUAL TO 'O' GO TO ER.
004211      GO TO A1.
004220 A.   MOVE POINTER2 TO POINTER1.
004221      MOVE LINK-OUT TO LINK.
004230 A1.  MOVE 16 TO RORW.
004240      MOVE 0 TO LENG.
004250      MOVE 67108873 TO SWIT.
004260      GO TO RTURN.
005010 TEST-BOOK.
005020      ADD 1 TO SUB.
005030      IF REPLY (SUB) IS NOT EQUAL TO 'O' GO TO ER.
005040      ADD 1 TO SUB.
005050      IF REPLY (SUB) IS NOT EQUAL TO 'O' GO TO ER.
005060      ADD 1 TO SUB.
005070      IF REPLY (SUB) IS NOT EQUAL TO 'K' GO TO ER.
            GO TO A1.
```

```
005090 TEST-JOUR.
            GO TO A1.
005110 TEST-DATE.
            ADD 1 TO SUB.
005130      IF REPLY (SUB) IS NOT EQUAL TO 'A' GO TO ER.
005140      ADD 1 TO SUB.
005150      IF REPLY (SUB) IS NOT EQUAL TO 'T' GO TO ER.
005160      ADD 1 TO SUB.
005170      IF REPLY (SUB) IS NOT EQUAL TO 'E' GO TO ER.
005180 MOD11C.
005181      MOVE 6 TO SUB. MOVE 1 TO SUB1.
005182 B.   MOVE REPLY (SUB) TO LOW-DATE (SUB1).
005183      ADD 1 TO SUB. ADD 1 TO SUB1.
005184      IF SUB1   4 GO TO C.
005185      GO TO B.
005186 C.   MOVE 1 TO SUB1. ADD 1 TO SUB.
005187 C1.  MOVE REPLY (SUB) TO HIGH-DATE (SUB1).
005188      ADD 1 TO SUB. ADD 1 TO SUB1.
005189      IF SUB1   4 GO TO D. GO TO C1.
005190 D.   MOVE 1 TO SUB SUB1.
005191 D1.  MOVE PTARAY1 (SUB) TO FORT-KEY.
005200      MOVE 4 TO RLDARN.
005210      ENTER LINKAGE.
005220      CALL 'IOENQ' USING RLDARN RTLIST DIRA FORT-KEY.
005230      ENTER COBOL.
005240      SUBTRACT 1 FROM COUNT-SAVE.
005241      SUBTRACT 1 FROM BLK-LEN.
005250      MOVE ACCES-NO TO DESC-SYM-KEY.
006010      MOVE ACCES-NO TO DESC-SEEK-KEY.
006020      MOVE 3 TO RLDARN.
006030      ENTER LINKAGE.
006040      CALL 'IOENQ' USING RLDARN RTLIST DIRA DESC-SYM-KEY.
006050      ENTER COBOL.
006060      IF IOERR = '1' GO TO END-LOOK.
006070      IF DATE-PUB1 ) L-DATE GO TO END-LOOK.
006080      IF DATE-PUB1   H-DATE GO TO END-LOOK.
```

```
006090      ADD 1 TO ALSO-COUNT.
006091      ADD 1 TO BLK-LEN-OUT.
006092      MOVE PTARAY1 (SUB) TO PTARAY2 (SUB).
006093      ADD 1 TO SUB1.
006100      IF COUNT-SAVE = 0 GO TO A1.
006110      IF BLK-LEN-OUT = 114 GO TO WRT-SCRT.
006160 END-LOOK.
006170      IF BLK-LEN = 0 GO TO RD-SCRT.
006180      ADD 1 TO SUB.
006190      GO TO D1.
006200 TEST-LANG.
            GO TO A1.
006220 WRT-SCRT.
006230      MOVE 17 TO RLDARN.
006240      ENTER LINKAGE.
006250      CALL 'IOENQ' USING RLDARN RTLIST POINTER2 LINK-OUT.
006260      ENTER COBOL.
006270      MOVE 0 TO BLK-LEN-OUT. MOVE 1 TO SUB1.
006280 WRT1.
006290      MOVE 0 TO PTARAY2 (SUB1).
007010      ADD 1 TO SUB1.
007020      IF SUB1   114 MOVE 1 TO SUB1 GO TO END-LOOK.
007030      GO TO WRT1.
007040 RD-SCRT.
007050      MOVE 16 TO RLDARN.
007060      ENTER LINKAGE.
007070      CALL 'IOENQ' USING RLDARN RTLIST POINTER1 LINK.
007080      ENTER COBOL.
007090      MOVE 1 TO SUB.
007100      GO TO D1.
```

```
001000 IDENTIFICATION DIVISION.
001010 PROGRAM-ID. 'M16'.
001020 AUTHOR. 'E.S.DAHL'.
000010 REMARKS.                                                         ABSTRACT
000020         ENTRY POINTS.                                            ABSTRACT
000030     'MOD16'.                                                     ABSTRACT
000040                                                                  ABSTRACT
000050     ACCESS.                                                      ABSTRACT
000060         SIMPLE SEARCH.                                           ABSTRACT
000070             SUBJECT.                                             ABSTRACT
000080                                                                  ABSTRACT
000090     INVOKED BY.                                                  ABSTRACT
000100         MODULE 5.                                                ABSTRACT
000110                                                                  ABSTRACT
000120     INVOKES.                                                     ABSTRACT
000130         MODULE 17 -                                              ABSTRACT
000140             IF SUBJECT IS IN THE DICTIONARY.                     ABSTRACT
000150         MODULE 11 -                                              ABSTRACT
000140             IF SUBJECT IS NOT IN THE DICTIONARY,                 ABSTRACT
000170                                                                  ABSTRACT
000180     SUBROUTINES.                                                 ABSTRACT
000190         IDENQ,                                                   ABSTRACT
000200         GOSYM,                                                   ABSTRACT
000210         SCPTXT.                                                  ABSTRACT
000220                                                                  ABSTRACT
000230     I/O ROUTINES.                                                ABSTRACT
000240         RDALP  (13),                                             ABSTRACT
000250         RDNUM  (14).                                             ABSTRACT
000260                                                                  ABSTRACT
000270     INPUT FILES.                                                 ABSTRACT
000280         ALPHABETIC DICTIONARY  (COB-ISAM),                       ABSTRACT
000290         NUMERIC    DICTIONARY  (COB-ISAM),                       ABSTRACT
000300                                                                  ABSTRACT
000310         MESSAGES.                                                ABSTRACT
000320         'THERE ARE N--N CITATIONS RELATIVE TO    '-              ABSTRACT
000330          REQUESTED SUBJECT NAME                                  ABSTRACT
```

```
000340          'APPEARING IN THE FOLLOWING GENERAL AREAS'-           ABSTRACT
000350           GENERAL CATEGORY NAME                                ABSTRACT
000370           GENERAL CATEGORY NAME                                ABSTRACT
000370           GENERAL CATEGORY NAME                                ABSTRACT
000370           GENERAL CATEGORY NAME                                ABSTRACT
000380          'ENTER NAME OF GENERAL AREA OF INTEREST  '.           ABSTRACT
000380                                                                ABSTRACT
000390      FUNCTION.                                                 ABSTRACT
000400          MODULE 16  SHOWS ON  THE 2260 THE NUMBER  OF  REFERENCES ABSTRACT
000410      SELECTED BY  MODULE 5 AND THE GENERAL CATEGORIES UNDER WHICH ABSTRACT
000420      THEY ARE CATALOGUED.   THE REQUESTED SUBJECT NAME IS USED TO ABSTRACT
000430      ACCESS  THE  'ALPHABETIC  FILE'  WHICH CONTAINS  A  LIST  OF ABSTRACT
000440      POINTERS TO THE 'NUMERIC FILE'  WHICH FURNISHES THE NAMES OF ABSTRACT
000450      THE REFERENCED CATEGORIES,  WHICH ARE  SHOWN  ON  THE  2260. ABSTRACT
000460      (ONLY THOSE  AT  THE MOST GENERAL LEVEL ARE SELECTED.)   THE ABSTRACT
000470      NAME  AND  NUMBER LISTS THUS  OBTAINED  ARE  STORED  IN  THE ABSTRACT
000480      DIRECT ACCESS BUFFER FOR USE BY MODULE 17.   MODULE 16  THEN ABSTRACT
000490      ASKS FOR THE NAME OF THE GENERAL AREA OF  INTEREST.   IF THE ABSTRACT
000500      REQUESTED CATEGORY IS NOT LISTED IN THE DICTIONARY,  CONTROL ABSTRACT
000510      IS PASSED DIRECTLY TO  MODULE 11.   IF  ONLY ONE CATEGORY IS ABSTRACT
000520      PERTINENT,  THE MESSAGE IS  LOADED INTO THE 2260 BUFFER LESS ABSTRACT
000530      THE  LAST LINE  BUT  IS  NOT  WRITTEN.,  CONTROL  IS  PASSED ABSTRACT
000540      DIRECTLY TO MODULE 17.                                    ABSTRACT
000
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 01  LOCAL-STORAGE.
001100     02  SUBJ            PICTURE X(80).
001102     02  NUMBERS.
001110     03  NUMBER-LIST                         OCCURS 5.
001120         04  NUMBER      PICTURE XXX.
001130         04  FILLER      PICTURE X(10).
```

```
001132     02  NAMES.
001140     03  NAME-LIST        PICTURE X(67)   OCCURS 5.
001150     02  IOPTRS                           COMPUTATIONAL.
001160         04  RDALP        PICTURE 9(8)    VALUE 13.
001170         04  RDNUM        PICTURE 9(8)    VALUE 14.
001180     02  CNTRS                            COMPUTATIONAL.
001190         04  LINEX        PICTURE 9(4)    VALUE 1.
001200         04  CHARX        PICTURE 9(4)    VALUE 1.
001210         04  LISTX        PICTURE 9(4)    VALUE 1.
001220         04  MOVEX        PICTURE 9(4)    VALUE 1.
001230     02  FILLER           PICTURE X(28).
001240     02  KOUNT            PICTURE 9(8)    COMPUTATIONAL.
002170 01  CIT-CNT.
002180     02  FILLER           PICTURE X(10)   VALUE
002190     'THERE ARE'.
002200     02  DEF-LOC.
002210         04  MASK         PICTURE ZZZ9.
002220         04  TEXT         PICTURE X(26)   VALUE
002230     ' CITATIONS RELATIVE TO    '.
002240     02  SHOUT1 REDEFINES DEF-LOC.
002250         04  FILLER       PICTURE X.
002260         04  SHFT1        PICTURE X(29).
002270     02  SHOUT2 REDEFINES DEF-LOC.
002280         04  FILLER       PICTURE XX.
002290         04  SHFT2        PICTURE X(28).
002300     02  SHOUT3 REDEFINES DEF-LOC.
002310         04  FILLER       PICTURE XXX.
002320         04  SHFT3        PICTURE X(27).
002330 01  WK.
002340     02  LOST             PICTURE 9(8)    COMPUTATIONAL.
002350     02  INT              PICTURE 9999.
002360     02  NUM-KEY.
002370         04  HI-ORDR      PICTURE XXX.
002380         04  FILLER       PICTURE X(13)   VALUE '.000.00.00   '.
003100 01  MSG11.
003110     02  DATXT.
```

```
003120          04  FILLER          PICTURE X(40)   VALUE
003130      'WOULD YOU CARE TO FURTHER DEFINE YOUR   '.
003140          04  FILLER          PICTURE X(40)   VALUE
003150      'SEARCH ACCORDING TO FOLLOWING OPTIONS   '.
003160          04  FILLER          PICTURE X(40)   VALUE
003170      'BOOKS ONLY                              '.
003180          04  FILLER          PICTURE X(40)   VALUE
003190      'JOURNALS ONLY                           '.
003200          04  FILLER          PICTURE X(40)   VALUE
003210      'DATE OF PUBLICATION YY TO YY            '.
003220          04  FILLER          PICTURE X(40)   VALUE
003230      'LANGUAGE                                '.
003240      02  OPTNS REDEFINES DATXT
003250                              PICTURE X(40)   OCCURS 6.
002382 LINKAGE SECTION.
002384 77   SWITCH              PICTURE 9(8)    COMPUTATIONAL.
002384 77   RTLIST              PICTURE 9(8)    COMPUTATIONAL.
002386 01   R8                                  COMPUTATIONAL.
002387      02  FILLER          PICTURE 9(8).
002388      02  RORW            PICTURE 9(8).
002389      02  LEN             PICTURE 9(8).
002390 01   MESAGE.
002400      02  FILLER          PICTURE X(10).
002410      02  CNTRS                           COMPUTATIONAL.
002420          04  FILLER      PICTURE 9(4).
002440          04  TOTAL       PICTURE 9(8).
002430          04  FILLER      PICTURE 9(8).
002450      02  FILLER          PICTURE X(460).
002460 01   SCOPE.
002470      02  FILLER          PICTURE X(4).
002480      02  TUBE1.
002490          04  SUBJECT     PICTURE X(80).
002500          04  FILLER      PICTURE X(400).
002510      02  TUBE2 REDEFINES TUBE1.
002520          04  SCOOP       PICTURE X(40)   OCCURS 12.
002530 01   DIRA                PICTURE X(524).
```

```
002540 01   ALP REDEFINES DIRA.
002550      02   ERRWD             PICTURE 9(8)     COMPUTATIONAL.
002570      02   FILLER            PICTURE X(68).
002580      02   CAT-KNT           PICTURE 9(4).
002590      02   STEP                               OCCURS 10.
002600           04  CAT-NO        PICTURE X(13).
002610           04  FILLER        PICTURE XXX.
003010 01   SAVE-DATA REDEFINES DIRA.
003020      02   SUBJ-HOLD         PICTURE X(80).
003030      02   NOS-HOLD          PICTURE X(65).
003040      02   NAMES-HOLD        PICTURE X(335).
003050      02   KOUNT-HOLD        PICTURE 9(8)     COMPUTATIONAL.
003010 01   NUM REDEFINES DIRA.
003020      02   ERRWD             PICTURE 9(8)     COMPUTATIONAL.
003040      02   CAT-NAME          PICTURE X(67).
004010 PROCEDURE DIVISION.
004020      ENTER LINKAGE.
004030      ENTRY 'MOD16' USING R8, RTLIST, MESAGE,
004040                         SCOPE, DIRA, SWITCH.
004050      ENTER COBOL.
004060      MOVE SUBJECT TO SUBJ.
004070      COMPUTE INT = TOTAL.
004090      MOVE INT TO MASK.
004090      IF INT ) '0010' MOVE SHFT3 TO DEF-LOC , GO TO S1.
004100      IF INT ) '0100' MOVE SHFT2 TO DEF-LOC , GO TO S1.
004110      IF INT ) '1000' MOVE SHFT1 TO DEF-LOC.
004120 S1.  MOVE CIT-CNT TO SCOOP (1).
004130      MOVE 80 TO LEN.
004140      ENTER LINKAGE.
004150      CALL 'SCPTXT' USING SUBJ, LEN, TUBE1, LINEX.
004160      ENTER COBOL.
004170      ADD 1 TO LINEX, MOVE
004180      'APPEARING IN THE FOLLOWING GENERAL AREAS' TO SCOOP (LINEX).
004190 GET-ALPHA.
004200      ENTER LINKAGE.
004210      CALL 'IOENQ' USING RDALP, RTLIST, ALP, SUBJ.
```

```
004220      ENTER COBOL.
004230      IF ERRWD OF ALP = 1 GO TO NO-ALP.
004240      IF CAT-KNT   5 MOVE 5 TO CAT-KNT.
005010      MOVE CAT-KNT TO KOUNT.
005020 L1.  MOVE CAT-NO (MOVEX) TO NUMBER-LIST (MOVEX).
005030      IF MOVEX = KOUNT GO TO GET-NUM.
005040      ADD 1 TO MOVEX, GO TO L1.
005050 GET-NUM.
005060      MOVE 1 TO MOVEX.
005070 N1.  MOVE NUMBER (MOVEX) TO HI-ORDR.
005080      ENTER LINKAGE.
005090      CALL 'IOENQ' USING RDNUM, RTLIST, NUM, NUM-KEY.
005100      ENTER COBOL.
005110      IF ERRWD OF NUM = 1 GO TO N2.
005112      MOVE CAT-NAME TO NAME-LIST (MOVEX).
005120      MOVE 67 TO LEN.
005130      ENTER LINKAGE.
005140      CALL 'SCPTXT' USING CAT-NAME, LEN, TUBE1, LINEX.
005150      ENTER COBOL.
005160      IF LINEX   10 MOVE 10 TO LINEX.
005170 N2.  IF MOVEX = KOUNT NEXT SENTENCE ELSE
005180      ADD 1 TO MOVEX, GO TO N1.
005181      MOVE 33554448 TO SWITCH.
005188      ADD 1 TO LINEX.
005186      IF KOUNT = 1 MOVE 16 TO RORW, MOVE LINEX TO LEN, GO TO RTRN.
005190      MOVE 'ENTER NAME OF GENERAL AREA OF INTEREST  '
005200      TO SCOOP (LINEX).
005220      COMPUTE LEN = LINEX * 40.
006010      MOVE 8 TO RORW.
006030      ENTER LINKAGE.
006040      CALL 'GOSYM' USING TUBE1, LEN.
006050      ENTER COBOL.
006060 RTRN.
006061      MOVE SUBJ   TO SUBJ-HOLD.
006062      MOVE NUMBERS TO NOS-HOLD.
006063      MOVE NAMES TO NAMES-HOLD.
```

```
006064     MOVE KOUNT TO KOUNT-HOLD.
006070     ENTER LINKAGE.
006080     RETURN.
006090     ENTER COBOL.
006100 NO-ALP.
006110     MOVE 1 TO MOVEX.
006120 MOD11.
006130     MOVE OPTNS (MOVEX) TO SCOOP (LINEX).
006140     ADD 1 TO MOVEX,
006150     IF MOVEX ) 7 ADD 1 TO LINEX, GO TO MOD11.
006160     COMPUTE LEN = LINEX * 40.
006170     ENTER LINKAGE.
006180     CALL 'GOSYM' USING TUBE1, LEN.
006180     ENTER COBOL.
006170     MOVE 8 TO RORW.
006130     MOVE 33554442 TO SWITCH.
006140     GO TO RTRN.
```

```
001000 IDENTIFICATION DIVISION.
001010 PROGRAM-ID. 'M17'.
001020 AUTHOR. 'E.S.DAHL'.
001030 ENVIRONMENT DIVISION.
001039 CONFIGURATION SECTION.
001040 SOURCE-COMPUTER. IBM-360.
001050 OBJECT-COMPUTER. IBM-360.
001060 DATA DIVISION.
001070 LINKAGE SECTION.
001080 77  SWITCH              PICTURE 9(8)    COMPUTATIONAL.
001090 77  RTLIST              PICTURE 9(8)    COMPUTATIONAL.
001100 01  R8                                  COMPUTATIONAL.
001110     02  FILLER          PICTURE 9(8).
001120     02  RORW            PICTURE 9(8).
001130     02  LEN             PICTURE 9(8).
001140 01  MESAGE.
001150     02  FILLER          PICTURE X(480).
002010 01  DIRA.
002020     02  SUBJECT         PICTURE X(80).
030        02  NUMBERS.
030040         04  NUMBER-LIST                 OCCURS 5.
030050             06  NUMBER  PICTURE XXX.
030060             06  FILLER  PICTURE X(10).
030070     02  NAMES.
030080         04  NAME-LIST                   OCCURS 5.
030090             06  NAME    PICTURE X(67).
030100     02  COUNTERS                        COMPUTATIONAL.
030110         04  KOUNT       PICTURE 9(8).
030120         04  J1          PICTURE 9(8).
030130         04  J2          PICTURE 9(8).
030140         04  J3          PICTURE 9(8).
030150         04  J4          PICTURE 9(8).
030150         04  I1          PICTURE 9(4).
030160         04  I2          PICTURE 9(4).
030170     02  KODE            PICTURE XXX.
030180     02  PNT             PICTURE X.
```

```
030190     02  PASS             PICTURE 9(4)    COMPUTATIONAL.
030200     02  UPORDN           PICTURE XX.
003010 01  SCOPE.
003020     02  SCOPE-LEN        PICTURE 9(8)    COMPUTATIONAL.
003030     02  REQUEST.
003040         04  SHORT-REQ    PICTURE X(40).
003050         04  RESTT-REQ    PICTURE X(40).
003060     02  FILLER           PICTURE X(320).
003070     02  TEST.
003080         04  TEST-LINE    PICTURE X(40)   OCCURS 2.
003090 01  ASDFLKJ REDEFINES SCOPE.
003090     02  FILLER           PICTURE XXXX.
003099     02  BUFFER.
003100         04  TEXT         PICTURE X(40)   OCCURS 12.
004010 PROCEDURE DIVISION.
004020     ENTER LINKAGE.
004030     ENTRY 'MOD17' USING R8, RTLIST, MESAGE,
004040                         SCOPE, DIRA, SWITCH.
004050     ENTER COBOL.
004055     IF RORW = 16 MOVE 1 TO I2, COMPUTE I1 = LEN, GO TO H1.
004060     MOVE 2 TO I1, MOVE 1 TO I2, MOVE 67 TO J4,
004070     IF SCOPE-LEN   40 ADD 1 TO I1.
004080     MOVE 'IS NOT AMONG AVAILABLE OPTIONS' TO TEXT (I1).
004090 ANAL.
004100     MOVE NAME (I2) TO TEST. MOVE 67 TO J4.
004110     ENTER LINKAGE.
004120     CALL 'SCPTXT' USING TEST, J4, BUFFER, I1.
004130     ENTER COBOL.
004140     IF J4 ) 41 GO TO S-COMP.
004150 L-COMP.
004160     SUBTRACT 1 FROM I1.
004170     MOVE TEXT (I1) TO TEST-LINE (1).
004180     ADD 1 TO I1,
004190     MOVE TEXT (I1) TO TEST-LINE (2).
004200     IF REQUEST = TEST GO TO HIT ELSE GO TO LOOP.
004210 S-COMP.
```

```
004220      IF SHORT-REQ = TEXT (I1) GO TO HIT.
005010 LOOP.
005020      ADD 1 TO I2, IF I2 NOT   KOUNT GO TO ANAL.
005040 NOHIT.
005050      ADD 1 TO I1.
005050      IF I1   10 MOVE 11 TO I1, ELSE ADD 1 TO I1.
005060      MOVE 'ENTER NAME OF GENERAL AREA OF INTEREST ' TO TEXT (I1).
005080      COMPUTE LEN = I1 * 40.
005090      MOVE 8 TO RORW.
005100      MOVE 33554448 TO SWITCH.
005110      GO TO RTRN.
005120 HIT.
005122      MOVE 1 TO I1.
005128 H1.
005130      MOVE NUMBER (I2) TO KODE.
005140      MOVE '.' TO PNT.
005150      MOVE                 'IS A MORE GENERAL OR A MORE SPECIFIC
005160      TO TEXT (I1), ADD 1 TO I1.
005170      MOVE                 'CATEGORY REQUIRED
005180      TO TEXT (I1).
005190      MOVE 8 TO RORW,
005200      MOVE 1 TO PASS,
005220      MOVE 33554449 TO SWITCH,
005230      COMPUTE LEN = I1 * 40.
006010 RTRN.
006020      ADD 1 TO I1.
006030      PERFORM CLRSCP VARYING I1 FROM I1 BY 1 UNTIL I1   12.
006040      ENTER LINKAGE.
            CALL 'GOSYM' USING BUFFER, LEN.
006060      ENTER LINKAGE.
006070      RETURN.
006080      ENTER COBOL.
006090 CLRSCP.
006100      MOVE SPACES TO TEXT (I1).
```

```
001000 IDENTIFICATION DIVISION.
001010 PROGRAM-ID. 'M18'.
00102- AUTHOR. 'E.S.DAHL'.
001030 ENVIRONMENT DIVISION.
00103  CONFIGURATION SECTION.
001040 SOURCE-COMPUTER. IBM-360.
001050 OBJECT-COMPUTER. IBM-360.
001060 DATA DIVISION.
001070 WORKING-STORAGE SECTION.
001080 01  ERR-MSG.
001090     02  L1              PICTURE X(40)   VALUE
001100                       'IS NOT A VALID RESPONSE                 '.
001120     02  L2              PICTURE X(40)   VALUE
001130                       'RESPOND WITH -                          '.
001140     02  L3              PICTURE X(40)   VALUE
001150                       '(MORE SPEC), (MORE GENL), OR (NEITHER)  '.
003100 01  MSG11.
003110     02  DATXT.
               04  FILLER          PICTURE S9(8)   COMPUTATIONAL.
003120         04  FILLER          PICTURE X(40)   VALUE
003130     'WOULD YOU CARE TO FURTHER DEFINE YOUR   '.
003140         04  FILLER          PICTURE X(40)   VALUE
003150     'SEARCH ACCORDING TO FOLLOWING OPTIONS   '.
003160         04  FILLER          PICTURE X(40)   VALUE
003170     'BOOKS ONLY                              '.
003180         04  FILLER          PICTURE X(40)   VALUE
003190     'JOURNALS ONLY                           '.
003200         04  FILLER          PICTURE X(40)   VALUE
003210     'DATE OF PUBLICATION YY TO YY            '.
003220         04  FILLER          PICTURE X(40)   VALUE
003230     'LANGUAGE                                '.
               04  FILLER          PICTURE XXXX    VALUE SPACES.
002010 LINKAGE SECTION.
002020 77 SWITCH               PICTURE 9(8)    COMPUTATIONAL.
00203  77  RTLIST              PICTURE 9(8)    COMPUTATIONAL.
002040 01  R8                                  COMPUTATIONAL.
```

```
002050      02  FILLER            PICTURE 9(8).
002060      02  RORW              PICTURE 9(8).
002070      02  LEN               PICTURE 9(8).
002080 01   MESAGE.
002090      02  FILLER            PICTURE X(480).
002100 01   DIRA.
002120      02  SUBJECT           PICTURE X(80).
002130      02  NUMBERS           PICTURE X(65).
002140      02  NAMES             PICTURE X(335).
002150      02  KOUNT             PICTURE XXXX.
002160      02  COUNTERS                              COMPUTATIONAL.
002120          04  J             PICTURE 9(8)        OCCURS 4.
002180          04  I1            PICTURE 9(4).
002190          04  I2            PICTURE 9(4).
002200      02  KODE              PICTURE XXXX.
002210      02  PASS              PICTURE 9(4)        COMPUTATIONAL.
002220      02  UPORDN            PICTURE XX.
002230      02  FILLER            PICTURE X(12).
003010 01   SCOPE.
003020      02  IN-LEN            PICTURE 9(8)        COMPUTATIONAL.
003030      02  IN-LINE.
003040          04  RESPONSE      PICTURE X(5)        OCCURS 2.
003050          04  FILLER        PICTURE X(30).
003060      02  OUT-LINE          PICTURE X(40)       OCCURS 11.
004010 PROCEDURE DIVISION.
004020      ENTER LINKAGE.
004030      ENTRY 'MOD18' USING R8, RTLIST, MESAGE,
004040                            SCOPE, DIRA, SWITCH.
004050      ENTER COBOL.
004060 VAL.
004070      MOVE 1 TO I1, I2.
004065      IF PASS = 2 GO TO ZERUT.
004080      IF  RESPONSE  (1) = 'MORE' ADD 1 TO I1.
004090      IF  RESPONSE (I1) = 'SPEC ' OR
&0410           RESPONSE (I1) = 'SPECI' MOVE 'DN' TO UPORDN, GO TO ZERUT.
&0411       IF  RESPONSE (I1) = 'GEN  ' OR
```

```
&04120          RESPONSE (I1) = 'GENL ' OR
&04130          RESPONSE (I1) = 'GENER' MOVE 'UP' TO UPORDN, GO TO ZERUT.
&04140      IF  RESPONSE (I1) = 'NO   ' OR
&04150          RESPONSE (I1) = 'SAME ' OR
&04160          RESPONSE (I1) = 'NEITH' GO TO MOD11.
&04170 ERRS.
004180      MOVE L1 TO OUT-LINE (1),
004190      MOVE L2 TO OUT-LINE (2),
004200      MOVE L3 TO OUT-LINE (3).
&04210      MOVE 160 TO LEN,
004220      MOVE 33554449 TO SWITCH,
004230      GO TO GOSYM.
005010 MOD11.
005020      MOVE MSG11 TO SCOPE.
005230      MOVE 33554442 TO SWITCH,
005040      MOVE 240 TO LEN.
005130 GOSYM.
005140      MOVE 8 TO RORW.
005150      ENTER LINKAGE.
            CALL 'GOSYM' USING IN-LINE, LEN.
005170      ENTER COBOL.
005180 RTRN.
005190      ENTER LINKAGE.
005200      RETURN.
005210      ENTER COBOL.
005050 ZERUT.
005070      PERFORM ZEROFF 4 TIMES.
005080      MOVE 0 TO I1, I2.
005090      MOVE J (1) TO IN-LEN.
005100      MOVE 16 TO RORW,
005110      MOVE 67108882 TO SWITCH,
005120      GO TO RTRN.
005220 ZEROFF. MOVE 0 TO J (I2), ADD 1 TO I2.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'M19'.
001030 AUTHOR. 'E.S.DAHL'.
001040 ENVIRONMENT DIVISION.
001045 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 01  LOCAL-STORAGE.
001100     02  SUBJECT         PICTURE X(80).
001110     02  NIMBERS.
001120         04  NUMBER-LIST                     OCCURS 5.
001130             06  NUMBER  PICTURE XXX.
001140             06  FILLER  PICTURE X(10).
001150     02  NAMES.
00116          04  NAME-LIST                       OCCURS 5.
001170             06  NAME    PICTURE X(67).
001180     02  COUNTERS                            COMPUTATIONAL.
001190         04  KJI.
001200             06  KOUNT   PICTURE 9(8).
001210             06  J1      PICTURE 9(8).
001220             06  J2      PICTURE 9(8).
001230             06  J3      PICTURE 9(8).
001240             06  J4      PICTURE 9(8).
001250         04  KJ  REDEFINES KJI.
001260             06  K       PICTURE 9(8).
001270             06  J       PICTURE 9(8)    OCCURS 4.
001280         04  IA.
001290             06  I1      PICTURE 9(4).
001300             06  I2      PICTURE 9(4).
001310         04  IB  REDEFINES IA.
001320             06  I       PICTURE 9(4)    OCCURS 2.
001330     02  KODE            PICTURE XXXX.
001340     02  PASS            PICTURE 9(4)    COMPUTATIONAL.
001350     02  UPORDN  PICTURE XX.
```

```
001360         88  UP                          VALUE 'UP'.
001370         88  DN                          VALUE 'DN'.
001382     02  P                   PICTURE S9(4)  COMPUTATIONAL
001383                                            VALUE ZERO.
001380     02  FILLER          PICTURE X(10).
001390 01  NUMBERS.
001400     02  NOSS                               OCCURS 5.
001410         04  NUM-LIST        PICTURE X(16).
001500  01 TOLL.
001510         02  DATXT.
001520             04  FILLER      PICTURE X(40)  VALUE
001530     'WOULD YOU CARE TO FURTHER DEFINE YOUR   '.
001540             04  FILLER      PICTURE X(40)  VALUE
001550     'SEARCH ACCORDING TO FOLLOWING OPTION    '.
001560             04  FILLER      PICTURE X(40)  VALUE
001570     'BOOKS ONLY                              '.
001580             04  FILLER      PICTURE X(40)  VALUE
001590     'JOURNALS ONLY                           '.
001600             04 FILLER       PICTURE X(40)   VALUE
001610     'DATE OF PUBLICATION YY TO YY            '.
001620             04  FILLER      PICTURE X(40)   VALUE
001630     'LANGUAGE                                '.
001640         02  OPTNS REDEFINES DATXT
001650                             PICTURE X(40)   OCCURS 6.
002010 LINKAGE SECTION.
002020 77  SWITCH              PICTURE 9(8)    COMPUTATIONAL.
002030 77  RTLIST              PICTURE 9(8)    COMPUTATIONAL.
002040 01  R8                                  COMPUTATIONAL.
002050     02  FILLER          PICTURE 9(8).
002060     02  RORW            PICTURE 9(8).
002070     02  LEN             PICTURE 9(8).
002080 01  MESAGE.
002090     02  FILLER          PICTURE X(480).
002100 01  DIRA.
002110     02  FILLER  PICTURE X(524).
002120 01  SCOPE.
```

```
002130      02  IN-LEN          PICTURE 9(8)    COMPUTATIONAL.
002140      02  SCOPE-TEXT.
002150          04  SCOOP       PICTURE X(40)   OCCURS 12.
003010 PROCEDURE DIVISION.
003020      ENTER LINKAGE.
003030      ENTRY 'MOD19' USING R8, RTLIST, MESAGE,
003031                          SCOPE, DIRA, SWITCH.
003040      ENTER COBOL.
003050 BOP.
003060      MOVE DIRA TO LOCAL-STORAGE.
003062      MOVE 0 TO P.
003070      IF J1 = 0 MOVE 0 TO I1 GO TO FORTRAN.
003072      IF SCOOP      (I1) = SPACES MOVE 0 TO I1 GO TO FORTRAN.
003081      ENTER LINKAGE.
003082      CALL 'CLRSCRX' USING RTLIST, MESAGE.
003083      ENTER COBOL.
003080      MOVE 16 TO RORW,
003090      MOVE SCOOP       (I1) TO SCOOP       (1),
003091          MOVE 40 TO IN-LEN,
003092          MOVE SPACES TO SCOOP       (2).
003095      IF I1   11 NEXT SENTENCE ELSE
003096          ADD 1 TO I1,
003097          MOVE SCOOP       (I1) TO SCOOP       (2),
003098          MOVE 80 TO IN-LEN.
003099      MOVE 67108868 TO SWITCH.
003100 RTRN.
003110      MOVE LOCAL-STORAGE TO DIRA.
003120      ENTER LINKAGE.
003130      RETURN.
003140      ENTER COBOL.
003150 FORTRAN.
003150      MOVE ALL ' ' TO SCOPE-TEXT.
003150      MOVE 0 TO I2.
003151      ADD 1 TO P, IF P   5 NEXT SENTENCE ELSE
003152      MOVE NUMBER-LIST (P) TO NUM-LIST (P) GO TO FORTRAN.
003153      ADD 1 TO J4.
```

```
003160      IF UPORDN = 'UP' GO TO UPFORT.
003170      ENTER LINKAGE.
003180      CALL 'DNTREE' USING NUMBERS, COUNTERS,
003190                          RTLIST, DIRA, SCOPE-TEXT.
003200      ENTER COBOL.
003210      GO TO FORT-BACK.
003220  UPFORT.
003221      MOVE 2 TO PASS, MOVE 99 TO I2
003230      ENTER LINKAGE.
003240      CALL 'UPTREE' USING NUMBERS, COUNTERS,
003250                          RTLIST, DIRA, SCOPE-TEXT.
003260      ENTER COBOL.
003270  FORT-BACK.
003261      IF I2 = 98, MOVE 1 TO I2, ADD 1 TO I1, GO TO MOD11.
003280      ADD 1 TO I1.
003290      IF I2 = 99 GO TO EOD.
003300      MOVE 'ENTER/SHIFT FOR MORE OR SELECT CATEGORY' TO SCOOP (I1).
003310      MOVE 18 TO I2 GO TO PUT-SCOPE.
004010  EOD.
004012      IF J4 = 1 MOVE 2 TO PASS.
004020      IF PASS = 1, MOVE
004030      'DO YOU WISH TO SEE THIS LIST AGAIN      ' TO SCOOP (I1),
004040      ELSE MOVE
004050      'DO YOU WISH TO REDEFINE YOUR SUBJECT    ' TO SCOOP (I1).
004060      MOVE 19 TO I2.
004069  PUT-SCOPE.
004070      COMPUTE LEN = I1 * 40.
004072      IF I2 = 18 COMPUTE SWITCH = 2 ** 24 + I2 ELSE
004080      COMPUTE SWITCH = 2 ** 25 + I2.
004090      MOVE 8 TO RORW.
004097      ENTER LINKAGE.
            CALL 'GOSYM' USING SCOPE-TEXT, LEN.
004099      ENTER COBOL.
003310      ADD 1 TO I1.
004100      GO TO RTRN.
005100   MOD11.
```

```
MOVE OPTNS (I2) TO SCOOP (I1).
IF I2 = 6 MOVE 10 TO I2 GO TO PUT-SCOPE.
ADD 1 TO I1, ADD 1 TO I2, GO TO MOD11.
```

```
001000 IDENTIFICATION DIVISION.
001010 PROGRAM-ID. 'M20'.
001020 AUTHOR. 'E. [illegible] DAHL'.
001030 ENVIRONMENT DIVISION.
001032 CONFIGURATION SECTION.
001040 SOURCE-COMPUTER. IBM-360.
001050 OBJECT-COMPUTER. IBM-360.
001060 DATA DIVISION.
003000 WORKING-STORAGE SECTION.
003100 01  MSG11.
003110     02  DATXT.
               04  FILLER          PICTURE S9(8)   COMPUTATIONAL.
003120         04  FILLER          PICTURE X(40)   VALUE
003130     'WOULD YOU CARE TO FURTHER DEFINE YOUR   '.
003140         04  FILLER          PICTURE X(40)   VALUE
003150     'SEARCH ACCORDING TO FOLLOWING OPTIONS   '.
003160         04  FILLER          PICTURE X(40)   VALUE
003170     'BOOKS ONLY                              '.
003180         04  FILLER          PICTURE X(40)   VALUE
003190     'JOURNALS ONLY                           '.
003200         04  FILLER          PICTURE X(40)   VALUE
003210     'DATE OF PUBLICATION YY TO YY            '.
003220         04  FILLER          PICTURE X(40)   VALUE
003230     'LANGUAGE                                '.
               04  FILLER          PICTURE XXXX    VALUE SPACES.
001070 LINKAGE SECTION.
001080 77  SWITCH              PICTURE 9(8)    COMPUTATIONAL.
001090 77  RTLIST              PICTURE 9(8)    COMPUTATIONAL.
001100 01  R8                                  COMPUTATIONAL.
001110     02  FILLER          PICTURE 9(8).
001120     02  RORW            PICTURE 9(8).
001130     02  LEN             PICTURE 9(8).
001140 01  MESAGE.
001150     02  FILLER          PICTURE X(480).
001160 01  DIRA.
001170     02  FILLER          PICTURE X(480).
```

```
001180     02  KOUNT           PICTURE 9(8)    COMPUTATIONAL.
001190     02  FILLER          PICTURE X(24).
001200     02  PASS            PICTURE 9(4)    COMPUTATIONAL.
001210     02  FILLER          PICTURE X(14).
001220 01  SCOPE.
001230     02  IN-LEN          PICTURE 9(8)    COMPUTATIONAL.
001240     02  SCOOP.
001250         04  ANS         PICTURE XXX.
001250         04  FILLER      PICTURE X(37).
001270     02  FILLER          PICTURE X(440).
002010 PROCEDURE DIVISION.
002020     ENTER LINKAGE.
002030     ENTRY 'MOD20' USING R8, RTLIST, MESAGE,
002040                         SCOPE, DIRA, SWITCH.
002050     ENTER COBOL.
002060     MOVE 8 TO RORW, MOVE 33554432 TO SWITCH.
002070     IF PASS = 1 GO TO PASS1.
002080 PASS2.
002090     IF ANS = 'NO ' COMPUTE SWITCH = SWITCH + 10,
002092     MOVE MSG11 TO SCOPE, MOVE 240 TO LEN, GO TO DO-IO.
002100     MOVE 40 TO LEN, ADD 4 TO SWITCH.
002130     ENTER LINKAGE.
002131     CALL 'CLRSCRX' USING RTLIST, MESAGE.
002131     ENTER COBOL.
002108     IF ANS NOT = 'YES' GO TO NO-IO.
002110     MOVE 'ENTER NAME OF CATEGORY OF INTEREST      ' TO SCOOP.
002131 DO-IO.
002132     ENTER LINKAGE.
           CALL 'GOSYM' USING SCOOP, LEN.
002150     ENTER COBOL.
002160     GO TO RTRN.
002170 PASS1.
002180     MOVE 0 TO IN-LEN, MOVE 2 TO PASS.
002190     COMPUTE SWITCH = SWITCH * 2 + 17.
002200     IF ANS = 'NO ' MOVE 0 TO KOUNT ADD 1 TO SWITCH.
002205 NO-IO.
```

```
002210     ADD 8 TO RORW.
002220 RTRN.
002230     ENTER LINKAGE.
002240     RETURN.
002250     ENTER COBOL.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'M50'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  RESULT                  PICTURE X.
001100 77  SUB                     PICTURE 9(3) COMPUTATIONAL.
001110 77  WHICH-ONE               PICTURE 9(6) COMPUTATIONAL.
001111 77  SUB1                    PICTURE 9(3) COMPUTATIONAL.
001112 01  ID.
001113     02  ID-HOLD             PICTURE X(10).
001120 LINKAGE SECTION.
001130 77  RTLIST                  PICTURE 9(6) COMPUTATIONAL.
001140 77  SWIT                    PICTURE 9(8) COMPUTATIONAL.
001150 01  REG8.
001160     02  FILLER              PICTURE X(4).
001170     02  RORW                PICTURE 9(6) COMPUTATIONAL.
001180     02  LENG                PICTURE 9(6) COMPUTATIONAL.
001190 01  SCOPE.
001200     02  FILLER              PICTURE X(4).
001210     02  TEXT                PICTURE X(480).
001220     02  FILLER              PICTURE X(440).
001230 01  SCOPE-OUT REDEFINES SCOPE.
001240     02  FILLER              PICTURE X(4).
001250     02  OUT-LINE            PICTURE X(480).
002010 01  MESAGE.
002020     02  ID-NO               PICTURE X(10).
002030     02  FILLER              PICTURE X(470).
002040 01  DIRA.
002050     02  DIRA1               PICTURE X(428).
002060 PROCEDURE DIVISION.
002070     ENTER LINKAGE.
002080     ENTRY 'MOD50' USING REG8 RTLIST MESAGE SCOPE DIRA SWIT.
```

```
002090       ENTER COBOL.
003060 C.    MOVE TEXT TO ID.
003070       ENTER LINKAGE.
003080       CALL 'IDLOOK' USING ID RESULT.
003090       ENTER COBOL.
             GO TO D.
003110       MOVE 'YOU ARE NOT A QUALIFIED USER, THE NEXT ATTEMPT TO ENTE
003120-      'R WILL INITIATE A WARNING SYSTEM TO THE HEAD LIBRARIAN. STOP
003130-      '.' TO OUT-LINE. MOVE 120 TO LENG.
003140       MOVE 8 TO RORW.
003150       MOVE 67108962 TO SWIT.
003160       ENTER LINKAGE.
003170       RETURN.
003180       ENTER COBOL.
003190 D.
003200       MOVE 'ENTER THE TITLE TO BE ORDERED, NOT MORE THAN 240 CHARAC
003210-      'TERS PLEASE.' TO OUT-LINE.
003220       MOVE 8 TO RORW.
003230       MOVE 33554482 TO SWIT.
003240       MOVE 80 TO LENG WHICH-ONE.
003250       ENTER LINKAGE.
004010       CALL 'GOSYM' USING OUT-LINE WHICH-ONE.
004020       ENTER COBOL.
004030       ENTER LINKAGE.
004040       RETURN.
004050       ENTER COBOL.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'LOOKID'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  SUB                     PICTURE 9(3) COMPUTATIONAL VALUE 1.
001100 77  NO1                     PICTURE X(10) VALUE '4525707773'.
001110 77  NO2                     PICTURE X(10) VALUE '4525628717'.
001120 LINKAGE SECTION.
001130 77  ID-NO                   PICTURE X(10).
001140 77  RESULT                  PICTURE X.
001150 PROCEDURE DIVISION.
001160     ENTER LINKAGE.
001170     ENTRY 'IDLOOK' USING ID-NO RESULT.
001180     ENTER COBOL.
001190     IF ID-NO = NO1 GO TO OK.
001200     IF ID-NO = NO2 GO TO OK.
001210     MOVE '0' TO RESULT.
001220 A.  ENTER LINKAGE.
001230     RETURN.
001240     ENTER COBOL.
001250 OK. MOVE '1' TO RESULT GO TO A.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'M51'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  WHICH-ONE                PICTURE 9(6) COMPUTATIONAL.
001100 LINKAGE SECTION.
001110 77  RTLIST                   PICTURE 9(6) COMPUTATIONAL.
001120 77  SWIT                     PICTURE 9(8) COMPUTATIONAL.
001130 01  REG8.
001140     02  FILLER               PICTURE X(4).
001150     02  RORW                 PICTURE 9(6) COMPUTATIONAL.
001160     02  LENG                 PICTURE 9(6) COMPUTATIONAL.
001170 01  MESAGE.
001180     02  ID-NO                PICTURE X(10).
001190     02  FILLER               PICTURE X(470).
001200 01  SCOPE.
001210     02  MES-LENG             PICTURE 9(6) COMPUTATIONAL.
001220     02  SCOPE-BUF            PICTURE X(240).
001221     02  FILLER               PICTURE X(240).
001230 01  DIRA.
001240     02  DIRA-TITLE           PICTURE X(240).
001250     02  FILLER               PICTURE X(188).
002010 PROCEDURE DIVISION.
002020     ENTER LINKAGE.
002030     ENTRY 'MOD51' USING REG8 RTLIST MESAGE SCOPE DIRA SWIT.
002040     ENTER COBOL.
002050     IF MES-LENG   240 GO TO A.
002060     MOVE SCOPE-BUF TO DIRA-TITLE.
002070     MOVE 'ENTER THE EDITION NUMBER 3 CHARACTERS.' TO SCOPE-BUF.
002080     MOVE 8 TO RORW.
002090     MOVE 33554483 TO SWIT.
002100     MOVE 40 TO LENG WHICH-ONE.
```

```
002110      ENTER LINKAGE.
002120      CALL 'GOSYM' USING SCOPE-BUF WHICH-ONE.
002130      ENTER COBOL.
002140      ENTER LINKAGE.
002150      RETURN.
002160      ENTER COBOL.
002170 A.
002180      MOVE 'YOUR TITLE CONTAINS MORE THAN 240       CHARACTERS, ENT
002190-     'ER TITLE AGAIN.' TO SCOPE-BUF.
002200      MOVE 8 TO RORW.
002210      MOVE 33554482 TO SWIT.
002220      MOVE 80 TO LENG WHICH-ONE.
002230      ENTER LINKAGE.
002240      CALL 'GOSYM' USING SCOPE-BUF WHICH-ONE.
002250      ENTER COBOL.
003010      ENTER LINKAGE.
003020      RETURN.
003030      ENTER COBOL.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'M52'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  WHICH-ONE                 PICTURE 9(6) COMPUTATIONAL.
001100 LINKAGE SECTION.
001110 77  RTLIST                    PICTURE 9(6) COMPUTATIONAL.
001120 77  SWIT                      PICTURE 9(8) COMPUTATIONAL.
001130 01  REG8.
001140         02  FILLER            PICTURE X(4).
001150         02  RORW              PICTURE 9(6) COMPUTATIONAL.
001160         02  LENG              PICTURE 9(6) COMPUTATIONAL.
001170 01  SCOPE.
001180         02  FILLER            PICTURE X(4).
001190         02  SCOPE-BUF.
001200             03  SCOPE-EDTN    PICTURE X(3).
001210             03  FILLER        PICTURE X(477).
001220 01 MESAGE.
001230         02  ID-NO             PICTURE X(10).
001240         02  FILLER            PICTURE X(470).
001250 01  DIRA.
002010         02  DIRA-TITLE        PICTURE X(240).
002020         02  DIRA-EDTN         PICTURE X(3).
002030         02  FILLER            PICTURE X(185).
002040 PROCEDURE DIVISION.
002050     ENTER LINKAGE.
002060     ENTRY 'MOD52' USING REG8 RTLIST MESAGE SCOPE DIRA SWIT.
002070     ENTER COBOL.
002080     MOVE SCOPE-EDTN TO DIRA-EDTN.
002090     MOVE 'ENTER PLACE OF PUBLICATION NOT MORE THAN120 CHARACTERS.
002100-    ' ' TO SCOPE-BUF.
002110     MOVE 8 TO RORW.
```

```
002120     MOVE 33554484 TO SWIT.
002130     MOVE 80 TO LENG WHICH-ONE.
002140     ENTER LINKAGE.
002150     CALL 'GOSYM' USING SCOPE-BUF WHICH-ONE.
002160     ENTER COBOL.
002170     ENTER LINKAGE.
002180     RETURN.
002190     ENTER COBOL.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'M53'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  WHICH-ONE               PICTURE 9(6) COMPUTATIONAL.
001091 77  RLDARN                  PICTURE 9(6) COMPUTATIONAL.
001121 77  SUB                     PICTURE 9(6).
001122 01  HOLDER.
001123     02  HOLD-IT             PICTURE X OCCURS 120 TIMES.
001100 LINKAGE SECTION.
001110 77  RTLIST                  PICTURE 9(6) COMPUTATIONAL.
001120 77  SWIT                    PICTURE 9(8) COMPUTATIONAL.
001130 01  REG8.
001140     02  FILLER              PICTURE X(4).
001150     02  RORW                PICTURE 9(6) COMPUTATIONAL.
001160     02  LENG                PICTURE 9(6) COMPUTATIONAL.
001170 01  SCOPE.
001180     02  MES-LENG            PICTURE 9(5) COMPUTATIONAL.
001190     02  SCOPE-BUF.
001200         03  SCOPE-PLACE     PICTURE X(120).
001210         03  FILLER          PICTURE X(360).
001220 01  MESAGE.
001230     02  ID-NO               PICTURE X(10).
001240     02  FILLER              PICTURE X(470).
001250 01  DIRA.
010010     02  FILLER              PICTURE X(244).
010020     02  DIRA-PLACE          PICTURE 9(6) COMPUTATIONAL.
010030     02  FILLER              PICTURE X(181).
010040 PROCEDURE DIVISION.
010050     ENTER LINKAGE.
010060     ENTRY 'MOD53' USING REG8 RTLIST MESAGE SCOPE DIRA SWIT.
010070     ENTER COBOL.
```

```
010080      IF MES-LENG   120 GO TO A.
010081      MOVE 20 TO RLDARN.
010082      ADD 1 TO MES-LENG.
010083      MOVE MES-LENG TO SUB.
010084      MOVE SCOPE-PLACE TO HOLDER.
010085      MOVE ',' TO HOLD-IT (SUB).
010086      MOVE HOLDER TO SCOPE-PLACE.
010090      ENTER LINKAGE.
010100      CALL 'IOENQ' USING RLDARN RTLIST DIRA-PLACE SCOPE-PLACE.
010110      ENTER COBOL.
010130      MOVE 8 TO RORW.
010140      MOVE 33554485 TO SWIT.
010150      MOVE 'ENTER PUBLISHER NOT MORE THAN 68 CHARACTERS.' TO
010151      SCOPE-BUF.
010160      MOVE 80 TO LENG WHICH-ONE.
010170      ENTER LINKAGE.
010180      CALL 'GOSYM' USING SCOPE-BUF WHICH-ONE.
010190      ENTER COBOL.
010200      ENTER LINKAGE.
010210      RETURN.
010220      ENTER COBOL.
010230 A.
010240      MOVE 'YOUR PLACE OF PUBLICATION CONTAINS MORE THAN
0200101     '120 CHARACTERS, ENTER AGAIN' TO SCOPE-BUF.
020020      MOVE 8 TO RORW.
020030      MOVE 33554484 TO SWIT.
020040      MOVE 80 TO LENG WHICH-ONE.
020050      ENTER LINKAGE.
020060      CALL 'GOSYM' USING SCOPE-BUF WHICH-ONE.
020070      ENTER COBOL.
020080      ENTER LINKAGE.
020090      RETURN.
020100      ENTER COBOL.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'M54'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77 WHICH-ONE                  PICTURE 9(6) COMPUTATIONAL.
001091 77  RLDARN                    PICTURE 9(6) COMPUTATIONAL.
001100 LINKAGE SECTION.
001110 77  RTLIST                    PICTURE 9(6) COMPUTATIONAL.
001120 77  SWIT                      PICTURE 9(8) COMPUTATIONAL.
001130 01  REG8.
001140      02  FILLER               PICTURE X(4).
001150      02  RORW                 PICTURE 9(6) COMPUTATIONAL.
001160      02  LENG                 PICTURE 9(6) COMPUTATIONAL.
001170 01  SCOPE.
001180      02  MES-LENG             PICTURE 9(5) COMPUTATIONAL.
001190      02  SCOPE-BUF.
001200          03 SCOPE-PUBLER      PICTURE X(68).
001210          03 FILLER            PICTURE X(412).
001220 01  MESAGE.
001230      02  ID-NO                PICTURE X(10).
001240      02  FILLER               PICTURE X(470).
001250 01  DIRA.
010010      02  FILLER               PICTURE X(248).
010020      02  DIRA-PUBLER          PICTURE 9(6) COMPUTATIONAL.
010030      02  FILLER               PICTURE X(177).
010040 PROCEDURE DIVISION.
010050      ENTER LINKAGE.
010060      ENTRY 'MOD54' USING REG8 RTLIST MESAGE SCOPE DIRA SWIT.
010070      ENTER COBOL.
010080      IF MES-LENG   68 GO TO A.
010081      MOVE 21 TO RLDARN.
010090      ENTER LINKAGE.
```

```
010100     CALL 'IOENQ' USING RLDARN RTLIST DIRA-PUBLER SCOPE-PUBLER.
010110     ENTER COBOL.
010130     MOVE 8 TO RORW.
010140     MOVE 33554486 TO SWIT.
010150     MOVE 'ENTER DATE OF PUBLICATION.' TO SCOPE-BUF.
010160     MOVE 40 TO LENG WHICH-ONE.
010170     ENTER LINKAGE.
010180     CALL 'GOSYM' USING SCOPE-BUF WHICH-ONE.
010190     ENTER COBOL.
010200     ENTER LINKAGE.
010210     RETURN.
010220     ENTER COBOL.
010230 A.
010240     MOVE 'YOUR PUBLISHER CONTAINS MORE THAN 68
010241-    'CHARACTERS, ENTER AGAIN' TO SCOPE-BUF.
020010     MOVE 8 TO RORW.
020020     MOVE 33554485 TO SWIT.
020030     MOVE 80 TO LENG WHICH-ONE.
020040     ENTER LINKAGE.
020050     CALL 'GOSYM' USING SCOPE-BUF WHICH-ONE.
020060     ENTER COBOL.
020070     ENTER LINKAGE.
020080     RETURN.
020090     ENTER COBOL.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'M55'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  WHICH-ONE                PICTURE 9(6) COMPUTATIONAL.
001100 LINKAGE SECTION.
001110 77  RTLIST                   PICTURE 9(6) COMPUTATIONAL.
001120 77  SWIT                     PICTURE 9(8) COMPUTATIONAL.
001130 01  REG8.
001140         02  FILLER           PICTURE X(4).
001150         02  RORW             PICTURE 9(6) COMPUTATIONAL.
001160         02  LENG             PICTURE 9(6) COMPUTATIONAL.
001170 01  SCOPE.
001180         02  FILLER           PICTURE X(4).
001190         02  SCOPE-BUF.
001200             03  PUB-DATE     PICTURE X(4).
001210         02  FILLER           PICTURE X(476).
001211 01  SCOPER REDEFINES SCOPE.
001212         02  FILLER           PICTURE X(4).
001213         02  SCOPER-BUF       PICTURE X(480).
001220 01  MESAGE.
001230         02  ID-NO            PICTURE X(10).
001240         02  FILLER           PICTURE X(470).
001250 01  DIRA.
010010         02  FILLER           PICTURE X(252).
010020         02  DIRA-DATE        PICTURE X(4).
010030         02  FILLER           PICTURE X(173).
010040 PROCEDURE DIVISION.
010050     ENTER LINKAGE.
010060     ENTRY 'MOD55' USING REG8 RTLIST MESAGE SCOPE DIRA SWIT.
010070     ENTER COBOL.
010080     MOVE PUB-DATE TO DIRA-DATE.
```

```
010090     MOVE 'ENTER ACCESSION NUMBER.' TO SCOPER-BUF.
010100     MOVE 8 TO RORW.
010110     MOVE 33554487 TO SWIT.
010120     MOVE 40 TO LENG WHICH-ONE.
010130     ENTER LINKAGE.
010140     CALL 'GOSYM' USING SCOPE-BUF WHICH-ONE.
010150     ENTER COBOL.
010160     ENTER LINKAGE.
010170     RETURN.
010180     ENTER COBOL.
```

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. 'M57'.
000030 AUTHOR. ED S DAHL.
010040 ENVIRONMENT DIVISION.
010050 DATA DIVISION.
010060 WORKING-STORAGE SECTION.
010200 01  ACCN-REC.
020010     02  ACCN-FLDS.
020020         04  ACCN-A              PICTURE X(16).
020030         04  ACCN-ALL            PICTURE X(24).
010070 01  DESC-REC.
010080     02  ERRBYTE                 PICTURE X.
010090     02  DESC-FLDS.
010098       03  ACCNO.
010100         04  ACCN-D              PICTURE X(10).
010110         04  EDNO                PICTURE X(3).
010112       03  STEP.
010120         04  DESC-ALL.
010130             06  FILLER          PICTURE X(31).
010140             06  DATE            PICTURE X(4).
010150             06  FILLER          PICTURE X(380).
010160             06  PNTR            PICTURE 9(6).
010170             06  PLACE           PICTURE 9(6).
010180             06  PUBLR           PICTURE 9(6).
010190             06  FILLER          PICTURE X(43).
020040 LINKAGE SECTION.
020050 77  RTLIST                      PICTURE 9(8)    COMPUTATIONAL.
020060 77  SWIT                        PICTURE 9(8)    COMPUTATIONAL.
020070 01  R8                                          COMPUTATIONAL.
020080     02  FILLER                  PICTURE 9(8).
020090     02  RORW                    PICTURE 9(8).
020100     02  LEN                     PICTURE 9(8).
020110 01  MESAGE.
020120     02  FILLER                  PICTURE X(480).
020130 01  SCOPE.
020140     02  INLEN                   PICTURE 9(8)    COMPUTATIONAL.
```

```
020150      02  SCOOP.
020160          04  AUTHOR              PICTURE X(213).
020170          04  FILLER              PICTURE X(3).
020180        03  DSECT.
020190          04  TITLE               PICTURE X(240).
020200          04  WHICH               PICTURE XXX.
030010          04  FILLER              PICTURE X.
030020          04  WHERE               PICTURE 9(8)    COMPUTATIONAL.
030030          04  WHO                 PICTURE 9(8)    COMPUTATIONAL.
030040          04  WHEN                PICTURE X(4).
030050          04  WHAT                PICTURE X(10).
030060        03  FILLER              PICTURE X.
030070 01   DIRA.
030080      02  FSECT                   PICTURE X(266).
030090      02  FILLER                  PICTURE X(258).
040010 PROCEDURE DIVISION.
040020      ENTER LINKAGE.
040030      ENTRY 'MOD57' USING R8, RTLIST, MESAGE,
040040                               SCOPE, DIRA, SWIT.
040050      ENTER COBOL.
040060 BOP.
040070      IF INLEN   213 MOVE
040080            'YOUR RESPONSE IS INCORRECT, TRY AGAIN' TO SCOOP,
040090             MOVE 40 TO LEN,
040100             MOVE 8 TO RORW,
040110             MOVE 33554488 TO SWIT,
040120             GO TO ERREXT.
040140 RWACN1.
040150      MOVE 5 TO LEN, MOVE 26 TO SWIT.
040160      ENTER LINKAGE.
040170      CALL 'IOENQ' USING SWIT, RTLIST, ERRBYTE, LEN.
040180      ENTER COBOL.
050010 BLD-RX.
050020      MOVE FSECT TO DSECT.
050030      MOVE WHAT TO ACCN-D.
050040      MOVE WHICH TO EDNO.
```

```
050050      MOVE ACCNO TO ACCN-A.
050060      MOVE SPACES TO DESC-ALL.
050070      MOVE LOW-VALUE TO ACCN-ALL.
050080      MOVE WHEN TO DATE.
050090      COMPUTE PNTR = LEN.
050100      COMPUTE PLACE = WHERE,
050110      COMPUTE PUBLR = WHO.
050120  WTACN.
050130      MOVE 32 TO SWIT.
050140      ENTER LINKAGE.
050150      CALL 'IDENQ' USING SWIT, RTLIST, ACCN-REC, LEN.
050160      ENTER COBOL.
050170  WDSCIO.
050180      MOVE 25 TO SWIT, MOVE 3 TO ERRBYTE.
060010      ENTER LINKAGE.
060020      CALL 'IDENQ' USING SWIT, RTLIST, DESC-REC, ACCN-A.
060030      ENTER COBOL.
060040  ADD-RTNS.
060050      ENTER LINKAGE.
060060      CALL 'ADDA' USING AUTHOR, ACCNO, DIRA, RTLIST.
060070      ENTER LINKAGE.
060080      CALL 'ADDT' USING TITLE, ACCNO, DIRA, RTLIST.
060090      ENTER COBOL.
060100  FINISH.
060110      MOVE 16 TO RORW,
060120      MOVE 40 TO LEN,
060130      COMPUTE SWIT = 33554432 + 99 - 1.
060140  ERREXT.
060150      ENTER LINKAGE.
060160      CALL 'GOSYM' USING SCOOP, LEN.
060170      ENTER LINKAGE.
060180      RETURN.
060190      ENTER COBOL.
```

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. 'M56'.
000030 AUTHOR. ED S DAHL.
010040 ENVIRONMENT DIVISION.
010050 CONFIGURATION SECTION.
010060 SOURCE-COMPUTER. IBM-360.
010070 OBJECT-COMPUTER. IBM-360.
010080 DATA DIVISION.
010090 LINKAGE SECTION.
010100 77  RTLIST                  PICTURE 9(8)    COMPUTATIONAL.
010110 77  SWIT                    PICTURE 9(8)    COMPUTATIONAL.
010120 01  R8                                      COMPUTATIONAL.
010130     02  FILLER              PICTURE 9(8).
010140     02  RORW                PICTURE 9(8).
010150     02  LEN                 PICTURE 9(8).
010160 01  MESAGE.
010170     02  FILLER              PICTURE X(480).
010180 01  SCOPE.
010190     02  INLEN               PICTURE 9(8)    COMPUTATIONAL.
010200     02  SCOOP.
010210         04  ACCN-IN         PICTURE X(10).
010220         04  FILLER          PICTURE X(470).
020010 01  DIRA.
020020     02  FILLER              PICTURE X(256).
020030     02  ACCN-NO             PICTURE X(10).
020040     02  FILLER              PICTURE X(258).
020050 PROCEDURE DIVISION.
020060     ENTER LINKAGE.
020070     ENTRY 'MOD56' USING R8, RTLIST, MESAGE,
020080                         SCOPE, DIRA, SWIT.
020088     ENTER COBOL.
020090 BOP.
020100     IF INLEN NOT = 10 MOVE
020110         'YOUR RESPONSE IS INCORRECT, TRY AGAIN' TO SCOOP,
020120          MOVE 55 TO SWIT, GO TO ERREXT.
020130     MOVE ACCN-IN TO ACCN-NO, MOVE
```

```
020140         'ENTER NAME OF AUTHOR, UP TO 213 CHARS' TO SCOOP.
020150          MOVE 56 TO SWIT.
020160 ERREXT.
020170     ADD 33554432 TO SWIT.
020180     MOVE 8 TO RORW.
020190     MOVE 40 TO LEN.
030010     ENTER LINKAGE.
030020     CALL 'GOSYM' USING SCOOP, LEN.
030030     ENTER LINKAGE.
030040     RETURN.
030050     ENTER COBOL.
```

```
PROGRAM-ID. 'M60'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01   GOSYLENT      PICTURE    9(6)     COMPUTATIONAL.
01   SCOPE.
     02  FILLER     PICTURE    XXXX.
     02  L1       PICTURE   X(120)   VALUE
     'PLEASE ENTER  ACCESSION NO.-DROP OR      RETAIN-DATE OF RECEI
     'PT-ACTUAL COST IN    ONE OF THE FOLLOWING FORMATS '.
     02  FILLER   PICTURE    X(40)    VALUE SPACES.
     02  L2       PICTURE    X(78)    VALUE
     'XXXXXXXXXX-MM/DD/YY-$$$.$$-RETAIN    OR XXXXXXXXXX-DROP'.
     02  STAR     PICTURE    XX.
LINKAGE SECTION.
01   RTLIST       PICTURE    9(6)  COMPUTATIONAL.
01   SWIT         PICTURE    9(8)  COMPUTATIONAL.
01   REG8.
     02  FILLER   PICTURE    X(4).
     02  RORW     PICTURE    9(6)  COMPUTATIONAL.
     02  LENG     PICTURE    9(6)  COMPUTATIONAL.
01   MESAGE       PICTURE    X.
01   DIRA         PICTURE    X.
01   SCOP         PICTURE    X(282).
PROCEDURE DIVISION.
START.
     ENTER LINKAGE.
     ENTRY 'MOD60' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
     ENTER COBOL.
101.
     MOVE 240 TO GOSYLENT.
     ENTER LINKAGE.
     CALL  'GOSYM'  USING  L1  GOSYLENT.
```

```
ENTER COBOL.
MOVE SCOPE TO SCOP.
MOVE 8 TO RORW.  MOVE 240 TO LENG. MOVE 33554492 TO SWIT.
ENTER LINKAGE.
RETURN.
ENTER COBOL.
```

```
PROGRAM-ID.   'M61'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.   IBM-360.
OBJECT-COMPUTER.   IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01   REL-KEY          PICTURE     9(6)    COMPUTATIONAL.
01   GOSYLENT    PICTURE   9(6)  COMPUTATIONAL.
01   SCOPE.
    02  FILLER     PICTURE    XXXX.
    02  L1.
        03  ACCESS-NO    PICTURE   X(10).
       03  FILLER      PICTURE   X.
       03  REDATE.
       04 MONTH    PICTURE  99.
        04  FILLER    PICTURE  X.
        04  DATE    PICTURE  99.
         04  FILLER    PICTURE  X.
        04  YEAR    PICTURE  99.
   03  DROPP  REDEFINES REDATE    PICTURE   X(8).
   03  FILLER      PICTURE  X.
   03  ACTUAL-COST.
        04  DOLLARS    PICTURE   999.
       04  FILLER     PICTURE  X.
        04  CENTS     PICTURE  99.
   03  ACTCST  REDEFINES ACTUAL-COST  PICTURE   X(6).
   03  FILLER      PICTURE  X(14).
   02  L2      PICTURE  X(120)   VALUE
   'THERE ARE FORMAT ERRORS IN THE ABOVE    DATA.'.
   02  L3      PICTURE    X(120)   VALUE
   'PLEASE RE-ENTER  ACCESSION NO.-DROP OR  RETAIN-DATE OF RECEI
   'PT-ACTUAL COST IN   ONE OF THE FOLLOWING FORMATS '.
   02  L4      PICTURE   X(80)    VALUE
   'XXXXXXXXXX-MM/DD/YY-$$$.$$-RETAIN    OR XXXXXXXXXX-DROP'.
   02  L5      PICTURE    X(80)     VALUE
```

```
    'OR IF YOU WANT TO TERMINATE, PLEASE     ENTER   /STOP'.
01  SYMBO-KEY     PICTURE  X(16).
01  RLDARN     PICTURE  9(6)  COMPUTATIONAL.
01  SW     PICTURE   9   VALUE  ZERO.
01  DIR-ACC.
    02  ACNO          PICTURE     X(16).
    02  IDX-TITL       PICTURE     9(6)  COMPUTATIONAL.
    02  FILLER        PICTURE     X(4).
    02  IDX-AUT       PICTURE     9(6)   COMPUTATIONAL.
    02  FILLER        PICTURE     X(12).
01  DIR-AUT.
    02  AUTH          PICTURE    X(213).
    02  FILLER          PICTURE    X(20).
01  DIR-TIT.
    02  TITL          PICTURE   X(240).
    02  FILLER          PICTURE   X(4).
01  AUTHO    PICTURE   X(213).
LINKAGE SECTION.
01  DIRA.
    02  HIER-COD    PICTURE X.
    02   R-W     PICTURE   X.
    02  FILLER    PICTURE  X(57).
    02  DAT-REC.
        03  MON   PICTURE  99.
        03  DAT  PICTURE  99.
        03  YR  PICTURE  99.
    02  ACTCOST.
        03  DOL   PICTURE  999.
       03  CNTS  PICTURE  99.
    02  FILLER     PICTURE   X(359).
    02  INDEX-ACC     PICTURE   9(6).
    02  FILLER     PICTURE      X(55).
01  RTLIST      PICTURE   9(6)  COMPUTATIONAL.
01  SWIT        PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE   X(4).
```

```
           02  RORW     PICTURE   9(6)  COMPUTATIONAL.
           02  LENG     PICTURE   9(6)  COMPUTATIONAL.
       01  MESAGE  PICTURE  X(480).
       01  SCOP.
           02  LEGTH      PICTURE      X(4).
           02  LL1      PICTURE   X(39).
           02  FILLER  PICTURE  X(441).
       PROCEDURE DIVISION.
       START.
           ENTER LINKAGE.
           ENTRY 'MOD61' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
           ENTER COBOL.
       2.  MOVE LL1 TO L1.
           IF  LEGTH  GREATER '0039'  GO TO ERR.
           IF MONTH NOT NUMERIC GO TO DROPT.
           IF DATE NOT NUMERIC GO TO DROPT.
           IF YEAR NOT NUMERIC GO TO DROPT.
       101.
           IF ACTCST EQUAL TO SPACES  GO TO UPDAT.
           IF DOLLARS NOT NUMERIC GO TO ERR.
           IF CENTS NOT NUMERIC GO TO ERR.
       102.
           MOVE 1 TO SW GO TO UPDAT.
       DROPT.  IF DROPP EQUAL TO 'DROP     '   MOVE 4 TO SW GO TO UPDAT.
       ERR.
           MOVE 440 TO GOSYLENT  LENG.
           ENTER LINKAGE.
           CALL 'GOSYM'  USING L1  GOSYLENT.
           ENTER COBOL.
       103.
           MOVE SCOPE TO SCOP.
           MOVE 8 TO RORW.   MOVE 33554492  TO SWIT.
       104.
           ENTER LINKAGE.
           RETURN.
           ENTER COBOL.
```

```
UPDAT.
    MOVE ACCESS-NO  TO SYMBO-KEY.
    MOVE '1' TO HIER-COD.
    MOVE 25  TO RLDARN.
105.
    ENTER LINKAGE.
    CALL  'IOENQ' USING RLDARN RTLIST DIRA SYMBO-KEY.
    ENTER COBOL.
106.
    IF HIER-COD EQUAL TO '1'  MOVE 2 TO SW  GO TO MESAGG.
    IF  SW  NOT EQUAL TO 4   GO TO 1.
       MOVE INDEX-ACC TO REL-KEY  MOVE 4 TO RLDARN.
107.
    ENTER LINKAGE.
    CALL 'IOENQ'  USING RLDARN RTLIST  DIR-ACC  REL-KEY.
    ENTER COBOL.
108.
    MOVE 5 TO RLDARN.
109.
    ENTER LINKAGE.
    CALL 'IOENQ' USING RLDARN RTLIST DIR-AUT IDX-AUT.
    ENTER COBOL.
110.
    ENTER LINKAGE.
    CALL  'DROPA'  USING AUTHO ACNO  DIRA  RTLIST.
    ENTER  COBOL.
111.
    MOVE 6 TO RLDARN.
    ENTER LINKAGE.
    CALL 'IOENQ'  USING RLDARN RTLIST DIR-TIT IDX-TITL.
    ENTER COBOL.
112.
    ENTER LINKAGE.
    CALL 'DROPT' USING TITL  ACNO  DIRA  RTLIST.
    ENTER COBOL.
113.
```

```
     MOVE HIGH-VALUE TO ACNO.   MOVE 32 TO RLDARN.
 114.
     ENTER LINKAGE.
     CALL 'IOENQ'  USING RLDARN RTLIST DIR-ACC REL-KEY.
     ENTER COBOL.
 115.
     MOVE HIGH-VALUE TO HIER-COD.
     MOVE 25  TO RLDARN.
 116.
     ENTER LINKAGE.
     CALL 'IOENQ'  USING RLDARN RTLIST DIRA SYMBO-KEY.
     ENTER COBOL.
 117.
     GO TO MESAGG.
 1.  MOVE MONTH TO MON MOVE DATE TO DAT  MOVE YEAR TO YR.
     IF SW EQUAL TO 1   MOVE DOLLARS TO DOL  MOVE CENTS TO CNTS.
 REWRT.    MOVE ACCESS-NO    TO SYMBO-KEY.
     MOVE '2' TO HIER-COD.
     MOVE 25 TO RLDARN.
 119.
     ENTER LINKAGE.
     CALL   'IOENQ'   USING RLDARN RTLIST DIRA SYMBO-KEY.
     ENTER COBOL.
 120.
     IF HIER-COD EQUAL TO '1'  MOVE 3 TO SW GO TO MESAGG.
 MESAGG.
     IF SW EQUAL TO 2  MOVE
     'NO RECORD CAN BE FOUND IN THE FILE BASED ON THE ABOVE DATA.'
                            TO L2  GO TO ERR.
 121.
     IF SW EQUAL TO 3  MOVE
     'REWRITE-OPERATION FAILED. THE ABOVE DATA IS NOT SUCCESSFULLY
-    'UPDATED.'  TO L2  GO TO ERR.
 122.
     MOVE 'THE APPROPRIATE RECORD IS SUCCESSFULLY  UPDATED BASED O
-    'N THE ABOVE DATA.'                          TO L2  GO TO ERR.
```

```
 PROGRAM-ID.  'M70'.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER.  IBM-360.
 OBJECT-COMPUTER.  IBM-360.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01  GOSYLENT     PICTURE    9(6)   COMPUTATIONAL.
 01  SCOPE.
     02  FILLER     PICTURE    XXXX.
     02  L1       PICTURE    X(120)   VALUE
     'DO YOU WANT TO UPDATE THE AUTHOR  PLEASEENTER YES-ACCESSION
-    'NO. OR NO IN THE    FOLLOWING FORMAT '.
     02  FILLER   PICTURE    X(40)    VALUE SPACES.
     02  L2  PICTURE   X(120)   VALUE
     'YES-XXXXXXXXXX       OR
-    '                      NO'.
 LINKAGE SECTION.
 01  RTLIST       PICTURE    9(6)  COMPUTATIONAL.
 01  SWIT         PICTURE    9(8)  COMPUTATIONAL.
 01  REG8.
     02  FILLER   PICTURE    X(4).
     02  RORW     PICTURE    9(6)  COMPUTATIONAL.
     02  LENG     PICTURE    9(6)  COMPUTATIONAL.
 01  MESAGE       PICTURE    X.
 01  DIRA         PICTURE    X.
 01  SCOP    PICTURE   X(284).
 PROCEDURE DIVISION.
     READY TRACE
 START.
     ENTER LINKAGE.
     ENTRY 'MOD70' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
     ENTER COBOL.
 101.
     MOVE    280   TO  GOSYLENT          LENG.
 102.
```

```
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
103.
    MOVE SCOPE TO SCOP.
    MOVE 8 TO RORW.                   MOVE 33554502 TO SWIT.
104.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
```

```
PROGRAM-ID.  'M71'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
01  SCOPE.
    02  FILLER    PICTURE    XXXX.
    02  L1       PICTURE    X(40).
    02  L2      PICTURE     X(120)      VALUE
    'SOME ERRORS WERE FOUND IN THE ABOVE      ANSWER. PLEASE RE-EN
    'TER YLS-ACCESSION   NO. OR NO IN THE FOLLOWING FORMAT '.
    02  L3      PICTURE     X(120)    VALUE
    'YES-XXXXXXXXXX     OR                       NO'.
01  SW      PICTURE   9   VALUE   ZERO.
LINKAGE SECTION.
01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE   X(4).
    02  RORW    PICTURE   9(6)  COMPUTATIONAL.
    02  LENG    PICTURE   9(6)  COMPUTATIONAL.
01  MESAGE       PICTURE   X(40).
01  DIRA         PICTURE   X.
01  SCOP.
    02  LEGTH     PICTURE     X(4).
    02  L11.
      03   NOL.
          04  NO   PICTURE   XX.
        04   FILLER    PICTURE   X.
       03    YES   REDEFINES   NOL   PICTURE   XXX.
      03    FILLER   PICTURE   X.
       03   ACN    PICTURE     X(10).
    02  FILLER     PICTURE   X(310).
```

```
PROCEDURE DIVISION.
     READY TRACE
     02  FILLER     PICTURE    X(40)    VALUE  SPACES.
START.
     ENTER LINKAGE.
     ENTRY 'MOD71' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
     ENTER COBOL.
101.
     MOVE  L11    TO  L1.
     IF  LEGTH  GREATER '0039'  GO TO 11.
102.
     IF YES NOT EQUAL TO 'YES' GO TO 10.
     IF  ACN  EQUAL TO SPACES  GO TO 11.
       MOVE ACN TO MESAGE MOVE 1 TO SW GO TO MESAGG.
10.  IF NO EQUAL TO 'NO' MOVE 2 TO SW  GO TO MESAGG.
11.  MOVE 33554502  TO SWIT.
ERR.
     MOVE    320   TO  GOSYLENT           LENG.
103.
     ENTER LINKAGE.
     CALL  'GOSYM'  USING  L1  GOSYLENT.
     ENTER COBOL.
104.
     MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
AA.
     ENTER LINKAGE.
     RETURN.
     ENTER COBOL.
MESAGG.
     IF  SW EQUAL TO 1  MOVE
     'DO YOU WANT TO ADD OR DROP THE AUTHOR   PLEASE ENTER
     '                     ADD       OR'  TO  L2  MOVE
     'DROP'   TO  L3  MOVE  33554503  TO  SWIT  GO TO ERR.
105.
     IF  SW  EQUAL TO  2  MOVE  33554507 TO SWIT
           MOVE 16 TO RORW GO TO AA.
```

```
 PROGRAM-ID.  'M72'.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER.  IBM-360.
 OBJECT-COMPUTER.  IBM-360.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
 01  SCOPE.
     02  FILLER    PICTURE   XXXX.
     02  L1      PICTURE   X(40)     VALUE   SPACES.
     02  FILLER    PICTURE   X(40)   VALUE  SPACES.
     02  L2      PICTURE   X(120)  VALUE
     'SOME ERRORS WERE FOUND IN THE ABOVE     ANSWER. PLEASE RE-EN
-    'TER                    ADD        OR'.
     02  L3     PICTURE     X(40)      VALUE
     'DROP'.
 LINKAGE SECTION.
 01  RTLIST      PICTURE   9(6)  COMPUTATIONAL.
 01  SWIT        PICTURE   9(8)  COMPUTATIONAL.
 01  REG8.
     02  FILLER  PICTURE   X(4).
     02  RORW    PICTURE   9(6)  COMPUTATIONAL.
     02  LENG    PICTURE   9(6)  COMPUTATIONAL.
 01  MESAGE.
     02    FILLER   PICTURE    X(10).
     02   A-D     PICTURE     X.
 01  DIRA        PICTURE   X.
 01  SCOP.
     02  LEGTH     PICTURE     X(4).
     02  L11.
         03   DROP     PICTURE     XXXX.
        03    AD   REDEFINES   DROP.
          04   ADDD     PICTURE     XXX.
        04   FILLER     PICTURE   X.
     02  FILLER    PICTURE   X(236).
```

```
PROCEDURE DIVISION.
    READY TRACE
START.
    ENTER LINKAGE.
    ENTRY 'MOD72' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
101.
    MOVE  L11   TO  L1.
    IF  LEGTH  GREATER '0039'  GO TO 11.
    IF  DROP EQUAL TO 'DROP'  MOVE  'D'  TO A-D  GO TO MESAGG.
102.
    IF  ADDD  EQUAL TO 'ADD'  MOVE  'A'  TO A-D  GO TO MESAGG.
11.  MOVE 33554503  TO SWIT.   MOVE  L11   TO  L1.
ERR.
    MOVE    240   TO  GOSYLENT          LENG.
103.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
104.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
MESAGG.
    MOVE  SPACE  TO L3.          MOVE
    'WHO IS THE AUTHOR      PLEASE ENTER
    '                    LAST-NAME, FIRST-NAME MIDDLE-NAME'
           TO L2  MOVE  33554504   TO  SWIT
        GO TO ERR.
```

```
PROGRAM-ID.  'M73'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01       ACCESSION-NO     PICTURE    X(16).
01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
01  SCOPE.
    02  FILLER    PICTURE   XXXX.
    02  L1       PICTURE   X(120)  VALUE
    'SOME ERRORS WERE FOUND IN THE ANSWER YOUENTERED. PLEASE RE-E
    'NTER '.
    02  L2       PICTURE   X(120)     VALUE
    'LAST-NAME, FIRST-NAME MIDDLE-NAME'.
    02   L3      PICTURE    X(120)   VALUE  SPACES.
    02  L4      PICTURE    X(40)    VALUE   SPACES.
01  RLDARN      PICTURE  9(6)  COMPUTATIONAL.
01  SW      PICTURE   9   VALUE  ZERO.
LINKAGE SECTION.
01  DIRA.
    02  HIER-COD    PICTURE X.
    02   R-W      PICTURE   X.
01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE   X(4).
    02  RORW    PICTURE   9(6)  COMPUTATIONAL.
    02  LENG    PICTURE   9(6)  COMPUTATIONAL.
01  MESAGE.
    02  AC-NO     PICTURE      X(10).
    02  A-D     PICTURE   X.
01  SCOP.
    02  LEGTH     PICTURE      X(4).
    02  AUTHO     PICTURE   X(213).
    02     FILLER    PICTURE     X(187).
PROCEDURE DIVISION.
    READY TRACE
START.
    ENTER LINKAGE.
    ENTRY 'MOD73' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
101.
    IF  LEGTH  GREATER '0213'  GO TO 11.
    MOVE AC-NO TO ACCESSION-NO.
    GO  TO  UPDAT.
11.  MOVE 33554504  TO SWIT.
ERR.
    MOVE    400   TO  GOSYLENT           LENG.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
102.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
UPDAT.
    IF  A-D  EQUAL TO 'D'  GO TO DRP.
    ENTER LINKAGE.
    CALL 'ADDA' USING AUTHO ACCESSION-NO DIRA RTLIST.
    ENTER COBOL.
103.
    MOVE 1 TO SW.   GO TO MESAGG.
DRP.
    ENTER LINKAGE.
    CALL 'DROPA'  USING AUTHO ACCESSION-NO DIRA RTLIST.
    ENTER COBOL.
104.
    IF HIER-COD NOT EQUAL TO '9'  MOVE 1 TO SW  GO TO MESAGG.
    MOVE  4  TO SW.
```

```
       MESAGG.
           IF SW EQUAL TO 1   MOVE
           'THE APPROPRIATE RECORD IS SUCCESSFULLY  UPDATED ACCORDING TO
      -    ' THE DATA YOU JUST  ENTERED.      DO YOU WANT TO UPDATE ANY
      -    ''  TO L1   MOVE
           'OTHER AUTHOR  PLEASE RE-ENTER YES-ACCESSION NO. OR NO IN THE
      -    ' FOLLOWING    FORMAT '  TO  L2     MOVE
           'YES-XXXXXXXXXX                         NO'  TO  L3
                   MOVE  33554502  TO SWIT  GO TO ERR.
        105.
           IF SW EQUAL TO 4  MOVE
           'NO RECORD CAN BE FOUND ACCORDING TO     THE DATA YOU JUST ENT
      -    'ERED. DO YOU WANTTO UPDATE ANY OTHER AUTHOR       PLEASE'
              TO L1  MOVE
           'RE-ENTER YES-ACCESSION NO. OR NO IN THE FOLLOWING FORMAT
      -    '                          YES-XXXXXXXXXX'  TO L2  MOVE
      -    ' NO'  TO L3   MOVE 33554502  TO SWIT GO TO ERR.
```

```
PROGRAM-ID.  'M74'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  SYMBO-KEY      PICTURE     X(16).
01  ACCESSION-NO      PICTURE     X(16).
01  GOSYLENT     PICTURE     9(6)     COMPUTATIONAL.
01  SCOPE.
    02  FILLER     PICTURE    XXXX.
    02  L1       PICTURE    X(120)   VALUE
    'SOME ERRORS WERE FOUND IN THE ANSWER YOUENTERED. PLEASE RE-E
    'NTER IT.'.
    02  L2       PICTURE    X(120)      VALUE   SPACES.
    02   L3      PICTURE     X(120)    VALUE   SPACES.
    02  L4      PICTURE     X(40)    VALUE   SPACES.
01  RLDARN      PICTURE      9(6)      COMPUTATIONAL.
01  SW      PICTURE   9    VALUE    ZERO.
LINKAGE SECTION.
01  DIRA.
    02  HIER-COD    PICTURE    X.
    02   R-W   PICTURE     X.
    02  FILLER      PICTURE      X(74).
    02  CALLNO      PICTURE      X(14).
01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE    X(4).
    02  RORW    PICTURE    9(6)  COMPUTATIONAL.
    02  LENG    PICTURE    9(6)  COMPUTATIONAL.
01  MESAGE.
    02  AC-NO      PICTURE     X(10).
01  SCOP.
    02  LEGTH      PICTURE      X(4).
    02  CALL-NO      PICTURE     X(14).
    02  FILLER      PICTURE     X(386).
PROCEDURE DIVISION.
START.
    ENTER LINKAGE.
    ENTRY 'MOD74' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
102.
    IF  LEGTH  GREATER '0039'  GO TO 11.
    MOVE AC-NO TO ACCESSION-NO.
    GO  TO  UPDAT.
11.  MOVE 33554505  TO SWIT.
ERR.
    MOVE    400    TO   GOSYLENT           LENG.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
104.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
105.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
UPDAT.
    MOVE AC-NO TO SYMBO-KEY.   MOVE 25 TO RLDARN.
    MOVE '1' TO HIER-COD.
106.
    ENTER LINKAGE.
    CALL 'IOENQ' USING RLDARN RTLIST DIRA SYMBO-KEY.
    ENTER COBOL.
    IF HIER-COD EQUAL TO '1'  MOVE 4  TO SW GO TO MESAGG.
107.
    MOVE CALL-NO TO CALLNO  MOVE 25 TO RLDARN.
    MOVE '2' TO HIER-COD.
    ENTER LINKAGE.
    CALL 'IOENQ' USING RLDARN RTLIST DIRA SYMBO-KEY.
```

```
           ENTER COBOL.
       MESAGG.
           IF SW EQUAL TO 4 MOVE
           'NO RECORD CAN BE FOUND ACCORDING TO THE DATA YOU JUST ENTERE
      -    'D. DO YOU WANT TO   UPDATE ANY OTHER CALL NUMBER  PLEASE '
                TO  L1   MOVE
           'RE-ENTER YES-ACCESSION NO. OR NO IN THE FOLLOWING FORMAT
      -    '                      YES-XXXXXXXXXX     OR'  TO L2  MOVE
      -    'NO'   TO L3   MOVE 33554517  TO SWIT GO TO ERR.
       109.
           MOVE
           'THE APPROPRIATE RECORD IS SUCCESSFULLY  UPDATED ACCORDING TO
      -    ' THE DATA YOU JUST  ENTERED. DO YOU WANT TO UPDATE ANY OTHER
      -    ''   TO L1   MOVE
           'CALL NUMBER    PLEASE RE-ENTER          YES-ACCESSION NO. OR
      -    ' NO IN THE FOLLOWINGFORMAT '  TO L2  MOVE
           '                                        YES-XXXXXXXXXX    OR
      -    '                  NO'  TO L3  MOVE  33554517 TO SWIT
                GO TO ERR.
```

```
 PROGRAM-ID. 'M75'.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER.  IBM-360.
 OBJECT-COMPUTER.  IBM-360.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
 01  SCOPE.
     02  FILLER    PICTURE   XXXX.
     02  L1       PICTURE   X(120)  VALUE
     'DO YOU WANT TO UPDATE THE THE NOTE     PLEASE ENTER YES-ACC
-    'ESSION NO. OR NO IN THE FOLLOWING FORMAT '.
    02  FILLER  PICTURE   X(40)   VALUE SPACES.
     02  L2  PICTURE  X(120)  VALUE
     'YES-XXXXXXXXXX     OR
-    '                    NO'.
 LINKAGE SECTION.
 01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
 01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
 01  REG8.
     02  FILLER  PICTURE   X(4).
     02  RORW    PICTURE   9(6)  COMPUTATIONAL.
     02  LENG    PICTURE   9(6)  COMPUTATIONAL.
 01  MESAGE      PICTURE   X.
 01  DIRA        PICTURE   X.
 01  SCOP        PICTURE   X(284).
 PROCEDURE DIVISION.
     ENTER LINKAGE.
     ENTRY 'MOD75' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
     ENTER COBOL.
     MOVE   280   TO  GOSYLENT          LENG.
     ENTER LINKAGE.
     CALL  'GOSYM'  USING  L1  GOSYLENT.
     ENTER COBOL.
     MOVE SCOPE TO SCOP. MOVE  8  TO  RORW. MOVE 33554522 TO SWIT.
```

```
ENTER LINKAGE.
RETURN.
ENTER COBOL.
```

```
   PROGRAM-ID.  'M76'.
   ENVIRONMENT DIVISION.
   CONFIGURATION SECTION.
   SOURCE-COMPUTER.  IBM-360.
   OBJECT-COMPUTER.  IBM-360.
   DATA DIVISION.
   WORKING-STORAGE SECTION.
   01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
   01  SCOPE.
       02  FILLER    PICTURE   XXXX.
       02  L1      PICTURE   X(120)  VALUE
       'DO YOU WANT TO UPDATE THE TITLE   PLEASEENTER YES-ACCESSION
-      'NO. OR NO IN THE    FOLLOWING FORMAT '.
       02  FILLER  PICTURE   X(40)   VALUE SPACES.
       02  L2  PICTURE  X(120)  VALUE
       'YES-XXXXXXXXXX      OR
-      '                    NO'.
   LINKAGE SECTION.
   01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
   01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
   01  REG8.
       02  FILLER   PICTURE   X(4).
       02  RORW     PICTURE   9(6)  COMPUTATIONAL.
       02  LENG     PICTURE   9(6)  COMPUTATIONAL.
   01  MESAGE       PICTURE   X.
   01  DIRA         PICTURE   X.
   01  SCOP         PICTURE   X(284).
   PROCEDURE DIVISION.
       READY TRACE
   START.
       ENTER LINKAGE.
       ENTRY 'MOD76' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
       ENTER COBOL.
   101.
       MOVE   280   TO  GOSYLENT          LENG.
       ENTER LINKAGE.
```

```
    CALL  'GDSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
102.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
    MOVE  33554508  TO SWIT.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
```

```
PROGRAM-ID.  'M77'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  GOSYLENT      PICTURE    9(6)     COMPUTATIONAL.
01  SCOPE.
     02  FILLER     PICTURE   XXXX.
     02  L1       PICTURE   X(40).
    02  FILLER  PICTURE   X(40)   VALUE SPACES.
     02  L2       PICTURE   X(120)  VALUE
     'SOME ERRORS WERE FOUND IN THE ABOVE     ANSWER. PLEASE RE-EN
     'TER YES-ACCESSION   NO. OR NO IN THE FOLLOWING FORMAT '.
     02  L3     PICTURE    X(120)    VALUE
     'YES-XXXXXXXXXX     OR                  NO'.
01  SW     PICTURE   9   VALUE   ZERO.
LINKAGE SECTION.
01  RTLIST        PICTURE   9(6)  COMPUTATIONAL.
01  SWIT          PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
     02  FILLER  PICTURE   X(4).
     02  RORW    PICTURE   9(6)  COMPUTATIONAL.
     02  LENG    PICTURE   9(6)  COMPUTATIONAL.
01  MESAGE        PICTURE   X(40).
01  DIRA          PICTURE   X.
01  SCOP.
     02  LEGTH      PICTURE      X(4).
     02  L11.
       03   NOL.
            04  NO   PICTURE   XX.
          04   FILLER    PICTURE   X.
         03    YES   REDEFINES   NOL   PICTURE   XXX.
        03    FILLER   PICTURE   X.
         03    ACN    PICTURE     X(10).
    02  FILLER      PICTURE   X(310).
PROCEDURE DIVISION.
START.
    ENTER LINKAGE.
    ENTRY 'MOD77' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
101.
    MOVE  L11   TO  L1.
    IF  LEGTH  GREATER '0039'  GO TO 11.
102.
    IF YES NOT EQUAL TO 'YES'  GO TO 10.
    IF  ACN  EQUAL TO SPACES  MOVE 1 TO SW GO TO MESAGG.
      MOVE ACN TO MESAGE MOVE 1 TO SW GO TO MESAGG.
10. IF NO EQUAL TO 'NO' MOVE 2 TO SW  GO TO MESAGG.
11.  MOVE 33554508  TO SWIT.
ERR.
    MOVE    320   TO  GOSYLENT          LENG.
104.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
105.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
AA.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
MESAGG.
    IF  SW EQUAL TO 1  MOVE
    ' DO YOU WANT TO ADD OR DROP THE TITLE    PLEASE ENTER
    '                        ADD       OR'   TO  L2  MOVE
    'DROP'   TO  L3  MOVE  33554509  TO  SWIT  GO TO ERR.
106.
    IF  SW  EQUAL TO  2  MOVE  33554512 TO SWIT
           MOVE 16 TO RORW GO TO AA.
```

```
PROGRAM-ID.  'M78'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  GOSYLENT      PICTURE    9(6)    COMPUTATIONAL.
01  SCOPE.
    02  FILLER    PICTURE   XXXX.
    02  L1       PICTURE   X(40)      VALUE   SPACES.
   02  FILLER  PICTURE   X(40)   VALUE SPACES.
    02  L2       PICTURE   X(120)  VALUE
    'SOME ERRORS WERE FOUND IN THE ABOVE      ANSWER. PLEASE RE-EN
    'TER                    ADD          OR'.
    02  L3      PICTURE    X(40)       VALUE
    'DROP'.
LINKAGE SECTION.
01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE   X(4).
    02  RORW    PICTURE   9(6)  COMPUTATIONAL.
    02  LENG    PICTURE   9(6)  COMPUTATIONAL.
01  MESAGE.
    02    FILLER   PICTURE    X(10).
    02   A-D     PICTURE     X.
01  DIRA         PICTURE   X.
01  SCOP.
    02  LEGTH     PICTURE     X(4).
    02  L11.
        03   DROP     PICTURE     XXXX.
       03   AD   REDEFINES   DROP.
         04   ADDD    PICTURE     XXX.
       04   FILLER     PICTURE   X.
    02  FILLER    PICTURE   X(236).
```

```
PROCEDURE DIVISION.
    READY TRACE
101.
    ENTER LINKAGE.
    ENTRY 'MOD78' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
102.
    MOVE  L11   TO  L1.
    IF  LEGTH  GREATER '0039'  GO TO 11.
103.
    IF  DROP EQUAL TO 'DROP'  MOVE  'D'  TO A-D  GO TO MESAGG.
106.
    IF  ADDD  EQUAL TO 'ADD'  MOVE  'A'  TO A-D  GO TO MESAGG.
11.  MOVE 33554509  TO SWIT.   MOVE  L11   TO  L1.
ERR.
    MOVE    240   TO  GOSYLENT          LENG.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
104.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
105.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
MESAGG.
    MOVE  SPACE  TO L3.          MOVE
    'WHAT IS THE TITLE    PLEASE ENTER IT.
    '
               TO L2  MOVE  33554510   TO  SWIT
            GO TO ERR.
    PROGRAM-ID.  'M79'.
    ENVIRONMENT DIVISION.
    CONFIGURATION SECTION.
    SOURCE-COMPUTER.  IBM-360.
    OBJECT-COMPUTER.  IBM-360.
    DATA DIVISION.
    WORKING-STORAGE SECTION.
    01  ACCESSION-NO     PICTURE    X(16).
    01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
    01  SCOPE.
        02  FILLER    PICTURE    XXXX.
        02  L1       PICTURE   X(120)   VALUE
        'SOME ERRORS WERE FOUND IN THE ANSWER YOUENTERED. PLEASE RE-E
-       'NTER IT.'.
        02  L2       PICTURE   X(120)     VALUE   SPACES.
        02   L3      PICTURE    X(120)   VALUE   SPACES.
        02  L4      PICTURE    X(40)   VALUE   SPACES.
    01  RLDARN     PICTURE     9(6)     COMPUTATIONAL.
    01  SW     PICTURE    9   VALUE    ZERO.
    LINKAGE SECTION.
    01  DIRA.
        02  HIER-COD   PICTURE    X.
        02   R-W   PICTURE     X.
    01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
    01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
    01  REG8.
        02  FILLER  PICTURE   X(4).
        02  RORW    PICTURE   9(6)  COMPUTATIONAL.
        02  LENG    PICTURE   9(6)  COMPUTATIONAL.
    01  MESAGE.
        02  AC-NO     PICTURE    X(10).
        02  A-D    PICTURE      X.
    01  SCOP.
        02  LEGTH     PICTURE     X(4).
        02  TITL    PICTURE    X(240).
        02   FILLER     PICTURE   X(160).
```

```
PROCEDURE DIVISION.
START.
    ENTER LINKAGE.
    ENTRY 'MOD79' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
    READY TRACE
101.
    IF  LEGTH  GREATER '0240'  GO TO 11.
    MOVE AC-NO TO ACCESSION-NO.
    GO  TO  UPDAT.
11.  MOVE 33554510  TO SWIT.
ERR.
    MOVE    400   TO  GOSYLENT          LENG.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
102.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
UPDAT.
    IF  A-D  EQUAL TO 'D'  GO TO DRP.
    ENTER LINKAGE.
    CALL 'ADDT' USING TITL  ACCESSION-NO DIRA RTLIST.
    ENTER COBOL.
103.
    MOVE 1 TO SW.   GO TO MESAGG.
DRP.
    ENTER LINKAGE.
    CALL 'DROPT'  USING TITL  ACCESSION-NO DIRA RTLIST.
    ENTER COBOL.
104.
    IF HIER-COD NOT EQUAL TO '9'  MOVE 1 TO SW  GO TO MESAGG.
    MOVE  4  TO SW.
MESAGG.
```

```
     IF SW EQUAL TO 1   MOVE
     'THE APPROPRIATE RECORD IS SUCCESSFULLY  UPDATED ACCORDING TO
-    ' THE DATA YOU JUST  ENTERED.       DO YOU WANT TO UPDATE ANY
-    ''  TO L1   MOVE
     'OTHER TITLE   PLEASE RE-ENTER YES-ACCESSION NO. OR NO IN THE
-    ' FOLLOWING FORMAT '     TO  L2     MOVE
     'YES-XXXXXXXXXX       OR                    NO'  TO  L3
             MOVE  33554508  TO SWIT  GO TO ERR.
 105.
     IF SW EQUAL TO 4  MOVE
     'NO RECORD CAN BE FOUND ACCORDING TO     THE DATA YOU JUST ENT
-    'ERED. DO YOU WANTTO UPDATE ANY OTHER TITLE         PLEASE'
        TO L1  MOVE
     'RE-ENTER YES-ACCESSION NO. OR NO IN THE FOLLOWING FORMAT
-    '                    YES-XXXXXXXXXX     OR'  TO L2  MOVE
-    'NO'   TO L3   MOVE 33554508  TO SWIT GO TO ERR.
```

```
       PROGRAM-ID.  'M81'.
       ENVIRONMENT DIVISION.
       CONFIGURATION SECTION.
       SOURCE-COMPUTER.  IBM-360.
       OBJECT-COMPUTER.  IBM-360.
       DATA DIVISION.
       WORKING-STORAGE SECTION.
       01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
       01  SCOPE.
           02  FILLER    PICTURE    XXXX.
           02  L1       PICTURE    X(120)  VALUE
           'DO YOU WANT TO UPDATE THE SUBJECT PLEASEENTER YES-ACCESSION
      -    'NO. OR NO IN THE    FOLLOWING FORMAT '.
          02  FILLER  PICTURE    X(40)   VALUE SPACES.
           02  L2  PICTURE  X(120)  VALUE
           'YES-XXXXXXXXXX      OR
      -    '                     NO'.
       LINKAGE SECTION.
       01  RTLIST       PICTURE    9(6)  COMPUTATIONAL.
       01  SWIT         PICTURE    9(8)  COMPUTATIONAL.
       01  REG8.
           02  FILLER  PICTURE    X(4).
           02  RORW    PICTURE    9(6)  COMPUTATIONAL.
           02  LENG    PICTURE    9(6)  COMPUTATIONAL.
       01  MESAGE       PICTURE    X.
       01  DIRA         PICTURE    X.
       01  SCOP         PICTURE    X(284).
       PROCEDURE DIVISION.
           READY TRACE
       START.
           ENTER LINKAGE.
           ENTRY 'MOD81' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
           ENTER COBOL.
       102.
           MOVE    280   TO  GOSYLENT          LENG.
           ENTER LINKAGE.
```

```
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
103.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW. MOVE 33554513 TO SWIT.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
```

```
PROGRAM-ID.  'M82'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  GOSYLENT      PICTURE    9(6)     COMPUTATIONAL.
01  SCOPE.
    02  FILLER     PICTURE    XXXX.
    02  L1       PICTURE    X(40).
   02  FILLER  PICTURE    X(40)    VALUE SPACES.
    02  L2       PICTURE    X(120)  VALUE
    'SOME ERRORS WERE FOUND IN THE ABOVE     ANSWER. PLEASE RE-EN
    'TER YES-ACCESSION   NO. OR NO IN THE FOLLOWING FORMAT '.
    02  L3      PICTURE     X(120)    VALUE
    'YES-XXXXXXXXXX      OR                    NO'.
01  SW     PICTURE   9   VALUE   ZERO.
LINKAGE SECTION.
01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE    X(4).
    02  RORW    PICTURE    9(6)  COMPUTATIONAL.
    02  LENG    PICTURE    9(6)  COMPUTATIONAL.
01  MESAGE       PICTURE   X(40).
01  DIRA         PICTURE   X.
01  SCOP.
    02  LEGTH      PICTURE      X(4).
    02  L11.
      03   NDL.
          04  NO   PICTURE   XX.
        04   FILLER     PICTURE   X.
       03    YES   REDEFINES   NOL   PICTURE   XXX.
      03    FILLER   PICTURE   X.
       03   ACN    PICTURE      X(10).
    02  FILLER     PICTURE   X(310).
PROCEDURE DIVISION.
    READY TRACE
START.
    ENTER LINKAGE.
    ENTRY 'MOD82' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
101.
    MOVE  L11   TO  L1.
    IF  LEGTH  GREATER '0039'  GO TO 11.
    IF YES NOT EQUAL TO 'YES'  GO TO 10.
102.
    IF  ACN  EQUAL TO SPACES  MOVE 1 TO SW GO TO MESAGG.
      MOVE ACN TO MESAGE MOVE 1 TO SW GO TO MESAGG.
10. IF NO EQUAL TO 'NO' MOVE 2 TO SW  GO TO MESAGG.
11.  MOVE 33554513  TO SWIT.
ERR.
    MOVE    320   TO  GOSYLENT          LENG.
103.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
AA.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
MESAGG.
    IF  SW EQUAL TO 1  MOVE
    ' DO YOU WANT TO ADD OR DROP THE SUBJECT  PLEASE ENTER
    '                      ADD      OR'   TO  L2  MOVE
    'DROP'   TO  L3  MOVE  33554514  TO  SWIT  GO TO ERR.
    IF  SW  EQUAL TO  2  MOVE  33554516 TO SWIT
          MOVE 16 TO RORW GO TO AA.
```

```
PROGRAM-ID.  'M83'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  GOSYLENT      PICTURE    9(6)    COMPUTATIONAL.
01  SCOPE.
    02  FILLER    PICTURE   XXXX.
    02  L1       PICTURE   X(40)     VALUE   SPACES.
   02  FILLER  PICTURE   X(40)   VALUE SPACES.
    02  L2       PICTURE   X(120)  VALUE
    'SOME ERRORS WERE FOUND IN THE ABOVE     ANSWER. PLEASE RE-EN
    'TER                  ADD          OR'.
    02  L3      PICTURE    X(40)      VALUE
    'DROP'.
LINKAGE SECTION.
01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE   X(4).
    02  RORW    PICTURE   9(6)  COMPUTATIONAL.
    02  LENG    PICTURE   9(6)  COMPUTATIONAL.
01  MESAGE.
    02    FILLER   PICTURE    X(10).
    02   A-D     PICTURE     X.
01  DIRA         PICTURE   X.
01  SCOP.
    02  LEGTH     PICTURE     X(4).
    02  L11.
        03   DROP      PICTURE      XXXX.
       03   AD   REDEFINES   DROP.
         04   ADDD     PICTURE      XXX.
       04   FILLER     PICTURE   X.
    02  FILLER    PICTURE   X(236).
```

```
PROCEDURE DIVISION.
    READY TRACE
START.
    ENTER LINKAGE.
    ENTRY 'MOD83' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
102.
    MOVE  L11   TO  L1.
    IF  LEGTH  GREATER '0039'  GO TO 11.
    IF  DROP EQUAL TO 'DROP'  MOVE  'D'  TO A-D  GO TO MESAGG.
103.
    IF  ADDD  EQUAL TO 'ADD'  MOVE  'A'  TO A-D  GO TO MESAGG.
11.  MOVE 33554514  TO SWIT.   MOVE  L11   TO  L1.
ERR.
    MOVE    240   TO  GOSYLENT          LENG.
104.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
105.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
MESAGG.
    MOVE  SPACE  TO L3.          MOVE
    'WHAT IS THE SUBJECT  PLEASE ENTER IT.
    '
              TO L2  MOVE  33554515   TO  SWIT
            GO TO ERR.
```

```
PROGRAM-ID.  'M84'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  ACCESSION-NO      PICTURE    X(16).
01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
01  SCOPE.
    02  FILLER    PICTURE    XXXX.
    02  L1      PICTURE   X(120)   VALUE
    'SOME ERRORS WERE FOUND IN THE ANSWER YOUENTERED. PLEASE RE-E
-   'NTER IT.'.
    02  L2      PICTURE   X(120)     VALUE   SPACES.
    02   L3     PICTURE    X(120)   VALUE  SPACES.
    02  L4     PICTURE    X(40)   VALUE  SPACES.
01  RLDARN     PICTURE     9(6)      COMPUTATIONAL.
01  SW     PICTURE   9   VALUE   ZERO.
LINKAGE SECTION.
01  DIRA .
    02  HIER-COD   PICTURE   X.
    02   R-W   PICTURE     X.
01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE   X(4).
    02  RORW    PICTURE   9(6)  COMPUTATIONAL.
    02  LENG    PICTURE   9(6)  COMPUTATIONAL.
01  MESAGE.
    02  AC-NO     PICTURE    X(10).
    02  A-D    PICTURE    X.
01  SCOP.
    02  LEGTH     PICTURE     X(4).
    02  SUBJ     PICTURE    X(240).
    02   FILLER     PICTURE   X(160).
```

```
PROCEDURE DIVISION.
    ENTER LINKAGE.
    ENTRY 'MOD84' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
    READY TRACE
102.
    IF  LEGTH  GREATER '0080'  GO TO 11.
    MOVE AC-NO TO ACCESSION-NO.
    GO  TO  UPDAT.
11.  MOVE 33554515  TO SWIT.
ERR.
    MOVE    400   TO  GOSYLENT          LENG.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
103.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
UPDAT.
    IF  A-D  EQUAL TO 'D'  GO TO DRP.
105.
    ENTER LINKAGE.
    CALL 'ADDS' USING SUBJ  ACCESSION-NO DIRA RTLIST.
    ENTER COBOL.
106.
    MOVE 1 TO SW.   GO TO MESAGG.
DRP.
    ENTER LINKAGE.
    CALL 'DROPS'  USING SUBJ  ACCESSION-NO DIRA RTLIST.
    ENTER COBOL.
    IF HIER-COD NOT EQUAL TO '9'  MOVE 1 TO SW  GO TO MESAGG.
    MOVE  4  TO SW.
MESAGG.
    IF SW EQUAL TO 1   MOVE
```

```
     'THE APPROPRIATE RECORD IS SUCCESSFULLY  UPDATED ACCORDING TO
-    ' THE DATA YOU JUST  ENTERED.      DO YOU WANT TO UPDATE ANY
-    ''  TO L1   MOVE
     'OTHER SUBJECT PLEASE RE-ENTER YES-ACCESSION NO. OR NO IN THE
-    ' FOLLOWING FORMAT '     TO  L2     MOVE
     'YES-XXXXXXXXXX      OR                     NO'  TO  L3
             MOVE  33554513  TO SWIT  GO TO ERR.
     IF SW EQUAL TO 4  MOVE
     'NO RECORD CAN BE FOUND ACCORDING TO     THE DATA YOU JUST ENT
-    'ERED. DO YOU WANTTO UPDATE ANY OTHER SUBJECT      PLEASE'
        TO L1  MOVE
     'RE-ENTER YES-ACCESSION NO. OR NO IN THE FOLLOWING FORMAT
-    '                    YES-XXXXXXXXXX     OR'  TO L2  MOVE
-    'NO'   TO L3   MOVE 33554513  TO SWIT GO TO ERR.
```

```
 PROGRAM-ID.  'M85'.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER.  IBM-360.
 OBJECT-COMPUTER.  IBM-360.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
 01  SCOPE.
     02  FILLER    PICTURE   XXXX.
     02  L1      PICTURE   X(120)  VALUE
     'DO YOU WANT TO UPDATE THE CALL NUMBER   PLEASE ENTER YES-ACC
-    'ESSION NO. OR NO IN THE FOLLOWING FORMAT '.
    02  FILLER  PICTURE   X(40)   VALUE SPACES.
     02  L2  PICTURE  X(120)  VALUE
     'YES-XXXXXXXXXX     OR
-    '                    NO'.
 LINKAGE SECTION.
 01  RTLIST      PICTURE   9(6)  COMPUTATIONAL.
 01  SWIT        PICTURE   9(8)  COMPUTATIONAL.
 01  REG8.
     02  FILLER  PICTURE   X(4).
     02  RORW    PICTURE   9(6)  COMPUTATIONAL.
     02  LENG    PICTURE   9(6)  COMPUTATIONAL.
 01  MESAGE      PICTURE   X.
 01  DIRA        PICTURE   X.
 01  SCOP        PICTURE   X(284).
 PROCEDURE DIVISION.
     ENTER LINKAGE.
     ENTRY 'MOD85' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
     ENTER COBOL.
     MOVE    280   TO  GOSYLENT          LENG.
     ENTER LINKAGE.
     CALL  'GOSYM'  USING  L1  GOSYLENT.
     ENTER COBOL.
     MOVE SCOPE TO SCOP. MOVE  8  TO  RORW. MOVE 33554517 TO SWIT.
```

```
ENTER LINKAGE.
RETURN.
ENTER COBOL.
```

```
PROGRAM-ID.  'M86'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
01  SCOPE.
    02  FILLER     PICTURE    XXXX.
    02  L1       PICTURE    X(40).
   02  FILLER  PICTURE    X(40)   VALUE SPACES.
    02  L2       PICTURE    X(120)   VALUE
    'SOME ERRORS WERE FOUND IN THE ABOVE     ANSWER. PLEASE RE-EN
    'TER YES-ACCESSION   NO. OR NO IN THE FOLLOWING FORMAT '.
    02  L3      PICTURE     X(120)     VALUE
    'YES-XXXXXXXXXX      OR                    NO'.
01  SW     PICTURE   9   VALUE   ZERO.
LINKAGE SECTION.
01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE    X(4).
    02  RORW    PICTURE    9(6)  COMPUTATIONAL.
    02  LENG    PICTURE    9(6)  COMPUTATIONAL.
01  MESAGE       PICTURE   X(40).
01  DIRA         PICTURE   X.
01  SCOP.
    02  LEGTH      PICTURE      X(4).
    02  L11.
      03   NOL.
          04  NO   PICTURE   XX.
        04   FILLER    PICTURE   X.
       03    YES   REDEFINES   NOL   PICTURE   XXX.
      03    FILLER   PICTURE   X.
       03    ACN     PICTURE     X(10).
```

```
          02  FILLER      PICTURE   X(310).
      PROCEDURE DIVISION.
          ENTER LINKAGE.
          ENTRY 'MOD86' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
          ENTER COBOL.
          MOVE  L11    TO  L1.
          IF  LEGTH  GREATER '0039'  GO TO 11.
          IF YES NOT EQUAL TO 'YES'  GO TO 10.
          IF  ACN  EQUAL TO SPACES  MOVE 1 TO SW GO TO MESAGG.
            MOVE ACN TO MESAGE MOVE 1 TO SW GO TO MESAGG.
      10. IF NO EQUAL TO 'NO' MOVE 2 TO SW  GO TO MESAGG.
      ERR.
          MOVE    320   TO  GOSYLENT          LENG.
          ENTER LINKAGE.
          CALL  'GOSYM'  USING  L1  GOSYLENT.
          ENTER COBOL.
          MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
          ENTER LINKAGE.
          RETURN.
          ENTER COBOL.
      MESAGG.
          IF  SW EQUAL TO 1  MOVE
          'PLEASE ENTER THE CALL NUMBER.'  TO L2 MOVE SPACES TO L3
                MOVE 33554505  TO SWIT  GO TO ERR.
          IF  SW  EQUAL TO  2  MOVE  33554506 TO SWIT
              MOVE SPACES TO L2 L3    GO TO ERR.
```

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. 'M87'.
000030 AUTHOR. ED S DAHL.
010040 ENVIRONMENT DIVISION.
010050 CONFIGURATION SECTION.
010060 SOURCE-COMPUTER. IBM-360.
010070 OBJECT-COMPUTER. IBM-360.
010080 DATA DIVISION.
010090 WORKING-STORAGE SECTION.
010100 01  RLDARN                                      COMPUTATIONAL.
010110     02  RDALP                   PICTURE 9(8)    VALUE   13.
010120     02  SU1FOR                  PICTURE 9(8)    VALUE   08.
010130     02  RWACN1                  PICTURE 9(8)    VALUE   26.
010140     02  WTSUB1                  PICTURE 9(8)    VALUE   28.
010150     02  WSUBIO                  PICTURE 9(8)    VALUE   24.
010160     02  FORTKEY                 PICTURE 9(8)    VALUE   0 .
010170     02  FORTRET                 PICTURE 9(8)    VALUE   0 .
010180     02  RDNXT                   PICTURE 9(8)    VALUE   33554432.
010190     02  WTNXT                   PICTURE 9(8)    VALUE   67108864.
010200     02  SUBSEQ                  PICTURE 9(8)    VALUE   2.
020010 LINKAGE SECTION.
020020 77  RTLIST                      PICTURE 9(8)    COMPUTATIONAL.
020030 77  SWIT                        PICTURE 9(8)    COMPUTATIONAL.
020040 01  R8                                          COMPUTATIONAL.
020050     02  FILLER                  PICTURE 9(8).
020060     02  RORW                    PICTURE 9(8).
020070     02  LEN                     PICTURE 9(8).
020080 01  MESAGE.
020090     02  ACNO                    PICTURE X(10).
020100     02  CNTRS                                   COMPUTATIONAL.
020110         04  H                   PICTURE 9(4).
020120         04  I                   PICTURE 9(8).
020130         04  J                   PICTURE 9(8).
020140     02  ARRAY                                   COMPUTATIONAL.
020150         04  PTR                 PICTURE 9(8)    OCCURS 115.
020160 01  DIRA.
```

```
020170     02  ALPHA-DICTY.
020180         04  ERRWD            PICTURE 9(8)    COMPUTATIONAL.
020190         04  FILLER           PICTURE X(236).
020199     02  NUMRFLR              PICTURE X(96).
020200     02  SUBJ1FLE.
020210         04  SUBJ1            PICTURE X(80).
020212         04  SUBJECT REDEFINES SUBJ1 PICTURE X(80).
020220         04  SUB2I            PICTURE 9(8)    COMPUTATIONAL.
020230         04  MESHNO-AND-FLR   PICTURE X(12).
030010     02  SUBJFLE.
030020         04  ERRBYTE          PICTURE X.
030030         04  SUBJ             PICTURE X(80).
030040         04  SUBJ1PTR         PICTURE 9(6).
030050 01  SCOPE.
030060     02  IN-LEN               PICTURE 9(8)    COMPUTATIONAL.
030070     02  SUBJEKT.
030080
030090         04  SKEEP            PICTURE X(4).
030100         04  FILLER           PICTURE X(76).
030110     02  SCOOP                PICTURE X(80)   OCCURS 5.
040010 PROCEDURE DIVISION.
040020     ENTER LINKAGE.
040030     ENTRY 'MOD87' USING R8, RTLIST, MESAGE,
040040                          SCOPE, DIRA, SWIT.
040050     ENTER COBOL.
040060     IF IN-LEN   80 GO TO ERREXT.
040080     IF SKEEP = 'KEEP' GO TO READ-SUBJ.
040082     MOVE SUBJEKT TO SUBJECT.
040100 READ-ALPHA.
040110     ENTER LINKAGE.
040120     CALL 'IOENQ' USING RDALP, RTLIST, DIRA, SUBJECT.
040130     ENTER COBOL.
040140     IF ERRWD = 1 GO TO ERREXT.
040150 READ-SUBJ.
040160     ENTER LINKAGE.
040170     CALL 'IOENQ' USING SUBSEQ, RTLIST, SUBJFLE, SUBJECT.
```

```
040180     ENTER COBOL.
040190     IF ERRBYTE NOT = '1' COMPUTE FORTKEY = SUBJ1PTR,
                   MOVE FORTKEY TO FORTRET,
040191             GO TO STORE-PTR.
040200     MOVE 4 TO FORTRET.
040202     ENTER LINKAGE.
040204     CALL 'IOENQ' USING RWACN1, RTLIST, ARRAY,  FORTRET.
040206     ENTER COBOL.
040208     MOVE FORTRET TO FORTKEY.
040210     COMPUTE SUBJ1PTR = FORTKEY,
040220     MOVE SUBJECT TO SUBJ,
040230     MOVE '3' TO ERRBYTE.
050010     ENTER LINKAGE.
050020     CALL 'IOENQ' USING WSUBIO, RTLIST, SUBJFLE, SUBJECT.
050030     ENTER COBOL.
050040     MOVE SUBJECT TO SUBJ1,
050050     MOVE 0 TO SUB2I,
050060     MOVE SPACES TO MESHNO-AND-FLR.
050065     MOVE FORTRET TO FORTKEY.
050070     ENTER LINKAGE.
050080     CALL 'IOENQ' USING WTSUB1, RTLIST, SUBJ1FLE, FORTKEY.
050090     ENTER COBOL.
050100 STORE-PTR.
050103     MOVE FORTRET TO PTR (1).
050110     MOVE 0 TO H, J, PTR (2), FORTRET
050120     MOVE 1 TO I,
050135     IF ERRWD NOT = 1 MOVE 8 TO FORTRET.
050140     COMPUTE SWIT = 79 + WTNXT + FORTRET.
050150     MOVE 16 TO RORW.
050160 RTRN.
050170     ENTER LINKAGE.
050180     RETURN.
050190     ENTER COBOL.
060010 ERREXT.
060020     IF IN-LEN   80,
060030         MOVE 'YOUR RESPONSE IS INCORRECT, TRY AGAIN' TO SCOOP (1)
```

```
060040          ELSE
060050          MOVE                'IS NOT A MESH HEADING, ENTER (KEEP) TO
060060-                             'CATALOG, OR ENTER AN ALTERNATE HEADING'
060070          TO SCOOP (1).
060090      MOVE 160 TO LEN.
060100      ENTER LINKAGE.
060110      CALL 'GOSYM' USING SUBJECT, LEN.
060120      ENTER COBOL.
060130      MOVE 8 TO RDRW,
060140      COMPUTE SWIT = 86 + RDNXT.
060150      GO TO RTRN.
```

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. 'M88'.
000030 AUTHOR. ED S DAHL.
010040 ENVIRONMENT DIVISION.
010050 CONFIGURATION SECTION.
010060 SOURCE-COMPUTER. IBM-360.
010070 OBJECT-COMPUTER. IBM-360.
010080 DATA DIVISION.
010085 WORKING-STORAGE SECTION.
010088 77  DUMBUF                    PICTURE X(80).
010090 01  RLDARN                                       COMPUTATIONAL.
010100     02  SU1FOR                PICTURE 9(8)    VALUE   08.
010110     02  RDALP                 PICTURE 9(8)    VALUE   13.
010120     02  RDNUM                 PICTURE 9(8)    VALUE   14.
010130     02  WSUBIO                PICTURE 9(8)    VALUE   24.
010140     02  RWACN1                PICTURE 9(8)    VALUE   26.
010150     02  WTSUB1                PICTURE 9(8)    VALUE   28.
010160     02  FORTKEY               PICTURE 9(8)    VALUE   0 .
010170     02  FORTRET               PICTURE 9(8)    VALUE   0 .
010180     02  RDNXT                 PICTURE 9(8)    VALUE   33554432.
010190     02  WTNXT                 PICTURE 9(8)    VALUE   67108864.
010200     02  SUBSEQ                PICTURE 9(8)    VALUE   2.
010210     02  LIST                  PICTURE 9(4)    VALUE   1.
020010     02  SIST                  PICTURE 9(4)    VALUE   0.
020020     02  CATS                  PICTURE 9(4)    VALUE   1.
020030     02  CEND                  PICTURE 9(4)    VALUE   0.
020040     02  I                     PICTURE 9(4)    VALUE   0.
020050     02  J                     PICTURE 9(4)    VALUE   0.
020060     02  K                     PICTURE 9(4)    VALUE   0.
020070     02  L                     PICTURE 9(4)    VALUE   0.
020080     02  M                     PICTURE 9(4)    VALUE   0.
020090     02  N                     PICTURE 9(4)    VALUE   0.
021011 01  BREAK-DOWN.
021021     02  REF-NUM.
021031         04  REF-BLD.
021041             06  LVLO          PICTURE XXX .
```

```
021051             06  FILLER       PICTURE X   .
021061             06  LVL1         PICTURE XXX .
021071             06  FILLER       PICTURE X   .
021081             06  LVL2         PICTURE XX  .
021091             06  FILLER       PICTURE X   .
021101             06  LVL3         PICTURE XX  .
021111         04  STAT             PICTURE X   .
021121         04  LEVL             PICTURE 9(4)    COMPUTATIONAL.
021131 01  NUMR-LIST.
021141     02  REF-NO                               OCCURS 115.
021151         04  REF-IND          PICTURE X(13).
021161         04  F-STAT           PICTURE X.
021171         04  F-LEVL           PICTURE 9(4)    COMPUTATIONAL.
030010 01  CAT-LIST.
030020     02  CAT-NO               PICTURE XXX     OCCURS 115.
030022     02  FILLER               PICTURE X(7).
030030 01  CAT-TABLE.
030040     02  CAT-HDR.
030050         04  CAT-NAME         PICTURE X(68)   OCCURS 8.
030055     02  CAT-TLR.
030060         04  CAT-CODE         PICTURE XXX     OCCURS 8.
030070 01  NUM-KEY.
030080     02  FILLER               PICTURE X(13).
030090     02  DRAG                 PICTURE XXX.
040010 LINKAGE SECTION.
040020 77  RTLIST                   PICTURE 9(8)    COMPUTATIONAL.
040030 77  SWIT                     PICTURE 9(8)    COMPUTATIONAL.
040040 01  R8                                       COMPUTATIONAL.
040050     02  FILLER               PICTURE 9(8).
040060     02  RORW                 PICTURE 9(8).
040070     02  LEN                  PICTURE 9(8).
040080 01  MESAGE.
040090     02  ACNO                 PICTURE X(10).
040100     02  CNTRS                                COMPUTATIONAL.
040110         04  CATYS            PICTURE 9(4).
040120         04  TOTAL            PICTURE 9(8).
```

```
040130          04  FILLER           PICTURE 9(8).
040140       02  ARRAY                                COMPUTATIONAL.
040150          04  PTR              PICTURE 9(8)     OCCURS 115.
040160 01    DIRA.
040165       02  BUFFER.
040170       03  ALPHAFLE.
040180          04  ERRWD-A          PICTURE 9(8)     COMPUTATIONAL.
040190          04  NAME             PICTURE X(68).
040200          04  KOUNT            PICTURE 9(4).
040210          04  CAT-PTR                           OCCURS 10.
040212              06  CAT-ID       PICTURE XXX.
040214              06  FILLER       PICTURE X(13).
040220          04  FILLER           PICTURE 9(8)     COMPUTATIONAL.
050010       03  NUMERFLE.
050020          04  ERRWD-N          PICTURE 9(8)     COMPUTATIONAL.
050030          04  NEXT-NAME        PICTURE X(68).
050040          04  FILLER           PICTURE X(24).
050050       03  SUBJ1FLE.
050060          04  SUBJ1            PICTURE X(80).
050070          04  SUB2I            PICTURE 9(8)     COMPUTATIONAL.
050080          04  MESHNO-AND-FLR   PICTURE X(12).
050090       03  SUBJFLE.
050100          04  ERRBYTE          PICTURE X.
050110          04  SUBJ             PICTURE X(80).
050120          04  SUBJ1PTR         PICTURE 9(6).
050130          03  FILLER           PICTURE X(107).
050140       02  SAVE REDEFINES BUFFER.
050150          04  RLHOLD           PICTURE X(64).
050160          04  CAT-TRAP.
050170              06  CAT-HOLD     PICTURE X(352).
050180              06  CDE-HOLD     PICTURE X(24).
060010 01    SCOPE.
060020       02  IN-LEN              PICTURE 9(8)     COMPUTATIONAL.
060030       02  SCOOP               PICTURE X(40)    OCCURS 12.
070010 PROCEDURE DIVISION.
070020 BOP.
```

```
070030     ENTER LINKAGE.
070040     ENTRY 'MOD88' USING R8, RTLIST, MESAGE,
070050                             SCOPE, DIRA, SWIT.
070060     ENTER COBOL.
070070 NOP. EXIT.
070080 START.
           MOVE NAME TO NEXT-NAME.
070090     PERFORM STORE-NOS THRU NOS-STORED.
070092     PERFORM GET-NUMERIC, PERFORM SET-LEVL.
070095     GO TO FRSTIME.
070100 GET-NUMERIC.
070110     MOVE REF-NO (LIST) TO REF-NUM,
070115     IF STAT = 'X' GO TO SHIFT-LEFT.
070117 NXT-LVL.
070120     GO TO RND1, RND2, RND3, DEPENDING LEVL.
070130 RND1. MOVE '000' TO LVL1.
070140 RND2. MOVE '00'  TO LVL2.
070150 RND3. MOVE '00'  TO LVL3.
070156     IF LEVL ) 0 GO TO SHIFT-LEFT.
070155     SUBTRACT 1 FROM LEVL,
070157     MOVE 'X' TO STAT, PERFORM S1.
070158     IF I NOT = 0 GO TO NXT-LVL.
070160     MOVE REF-NUM TO NUM-KEY,
070170     MOVE SPACES TO DRAG.
070180     ENTER LINKAGE.
070190     CALL 'IDENQ' USING RDNUM, RTLIST, NUMERFLE, NUM-KEY.
070200     ENTER COBOL.
080010     IF ERRWD-N = 1 GO TO NXT-LVL.
080020 GET-ALPHA.
080030     ENTER LINKAGE.
080040     CALL 'IDENQ' USING RDALP, RTLIST, ALPHAFLE, NEXT-NAME.
080050     ENTER COBOL.
080060     IF ERRWD-A = 1 GO TO GET-NUMERIC.
081010 STORE-NOS.
081020     MOVE KOUNT TO N, MOVE 1 TO J.
081030 LOAD-NO.
```

```
081040      IF J   N GO TO NOS-STORED.
081050      MOVE CAT-PTR (J) TO REF-NUM,
081060      MOVE SPACE TO STAT.
081070 SET-LEVL.
081080      IF LVL3 NOT = '00'  MOVE 3 TO LEVL GO TO S.
081090      IF LVL2 NOT = '00'  MOVE 2 TO LEVL GO TO S.
081100      IF LVL1 NOT = '000' MOVE 1 TO LEVL GO TO S.
081110                          MOVE 0 TO LEVL.
081115 S.   EXIT.
081120 SS.  PERFORM S1.
081130      ADD 1 TO J, GO TO LOAD-NO.
081140 S1.  PERFORM NOP VARYING I FROM 1 BY 1 UNTIL
081150          I   SIST OR REF-BLD = REF-IND (I).
081160      IF  I   SIST MOVE I TO SIST, MOVE 0 TO I,
081170          IF  SIST   115 GO TO ABEND ELSE
081180          MOVE REF-NUM TO REF-NO (SIST).
081190 NOS-STORED. EXIT.
090010 READ-SUBJ.
090016      MOVE NEXT-NAME TO SUBJ1.
090020      ENTER LINKAGE.
090030      CALL 'IOENQ' USING SUBSEQ, RTLIST, SUBJFLE, SUBJ1.
090040      ENTER COBOL.
090050      IF ERRBYTE NOT = '1' GO TO STORE-PTR.
090060      MOVE 4 TO FORTRET.
090070      ENTER LINKAGE.
090080      CALL 'IOENQ' USING RWACN1, RTLIST, DUMBUF, FORTRET.
090090      ENTER COBOL.
090100      MOVE FORTRET TO FORTKEY, COMPUTE SUBJ1PTR = FORTKEY.
090110      MOVE NEXT-NAME TO SUBJ.
090120      MOVE '3' TO ERRBYTE.
090130      ENTER LINKAGE.
090140      CALL 'IOENQ' USING WSUBIO, RTLIST, SUBJFLE, SUBJ1.
090150      ENTER COBOL.
090170      MOVE 0 TO SUB2I,
090180      MOVE SPACES TO MESHNO-AND-FLR.
090190      ENTER LINKAGE.
```

```
090200     CALL 'IOENQ' USING WTSUB1, RTLIST, SUBJ1FLE, FORTKEY.
090210     ENTER COBOL.
100010 STORE-PTR.
100020     PERFORM NOP VARYING I FROM 1 BY 1 UNTIL
100022          I   TOTAL OR SUBJ1PTR = PTR (I).
100024     IF   I   TOTAL MOVE I TO TOTAL MOVE 0 TO I,
100026          IF  TOTAL   114 GO TO ABEND ELSE,
100028         MOVE SUBJ1PTR TO PTR (TOTAL).
100035 FRSTIME.
100040     MOVE LVL0 TO CAT-NO (TOTAL).
100050     IF LEVL NOT = 0 GO TO GET-NUMERIC.
100060 STORE-CAT.
100070     PERFORM NOP VARYING I FROM 1 BY 1 UNTIL
100080          I   CEND OR NEXT-NAME = CAT-NAME (I).
100090     IF   I   CEND MOVE I TO CEND MOVE 0 TO I,
100100          IF  CEND   8 GO TO ABEND ELSE,
100110         MOVE NEXT-NAME TO CAT-NAME (CEND).
100115         MOVE LVL0 TO CAT-CODE (CEND).
100120 SHIFT-LEFT.
100130     ADD 1 TO LIST.
100140     IF LIST NOT   SIST GO TO GET-NUMERIC.
110010 FINISH.
110020     COMPUTE I = IN-LEN / 40 + 2.
110030     MOVE 1 TO J.
110040     MOVE 'APPEARS IN THE FOLLOWING GENERAL AREA/S-' TO SCOOP (I).
110050 PUT-CATS.
110060     IF J   CEND GO TO SAVE-DATA.
110070     MOVE CAT-NAME (J) TO NAME.
110080     MOVE 67 TO L.
110090     ENTER LINKAGE.
110100     CALL 'SCPTXT' USING NAME, K, SCOOP (1), I.
110110     ENTER COBOL.
110120     IF I ) 11 ADD 1 TO J GO TO PUT-CATS.
110130 SAVE-DATA.
110135     ADD 1 TO I
110140     MOVE 'ENTER NAME OF ANY CATEGORIES TO DELETE' TO SCOOP (I).
```

```
110150      COMPUTE LEN = I * 40.
110160      ENTER LINKAGE.
110170      CALL 'GOSYM' USING SCOOP (1), LEN.
110180      ENTER COBOL.
110190      MOVE 8 TO RORW.
110200      COMPUTE SWIT = RDNXT + 88.
120010      MOVE CEND TO CATYS.
120020      MOVE CAT-TLR TO CDE-HOLD,
120030      MOVE CAT-LIST TO CAT-HOLD.
120040  RTRN.
120050      ENTER LINKAGE.
120060      RETURN.
120070      ENTER COBOL.
120080  ABEND.
120090      MOVE 16 TO RORW.
120100      COMPUTE SWIT = WTNXT + 98.
120110      GO TO RTRN.
```

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. 'M89'.
0/0030 AUTHOR. ED S DAHL.
010040 ENVIRONMENT DIVISION.
010050 CONFIGURATION SECTION.
010060 SOURCE-COMPUTER. IBM-360.
010070 OBJECT-COMPUTER. IBM-360.
010080 DATA DIVISION.
010090 WORKING-STORAGE SECTION.
010100 01  SUBS                                   COMPUTATIONAL.
010110     02  I               PICTURE 9(4)       VALUE 0.
010120     02  J               PICTURE 9(4)       VALUE 1.
010130     02  K               PICTURE 9(4)       VALUE 0.
010140     02  L               PICTURE 9(4)       VALUE  80.
010150     02  M               PICTURE 9(4)       VALUE  0.
010160     02  N               PICTURE 9(4)       VALUE  0.
010170     02  V               PICTURE 9(4)       OCCURS 8.
010180     02  RDNUM           PICTURE 9(8)       VALUE  14.
010190     02  RDNXT           PICTURE 9(8)       VALUE  33554432.
010200     02  WTNXT           PICTURE 9(8)       VALUE  67108864.
020010 01  CAT-KEY.
020020     02  CAT-NO          PICTURE XXX.
020030     02  FILLER          PICTURE X(13)      VALUE
020040                         '.000.00.00    '.
020050 01  CAT-NAMES.
020060     02  CAT-NAME                           OCCURS 3.
020070         04  CAT-CHAR    PICTURE X          OCCURS 80.
020080 01  CAT-ENTRY.
020090     02  CONSTANT.
020100         04  FILLER      PICTURE X          OCCURS 80
020110                                            DEPENDING L.
020110 01  CAT-INP.
020120     02  ERRWD           PICTURE 9(8)       COMPUTATIONAL.
020130     02  CAT-TAG         PICTURE X(68).
020140     02  FILLER          PICTURE  X(20).
030010 LINKAGE SECTION.
```

```
030020 77  RTLIST                       PICTURE 9(8)    COMPUTATIONAL.
030030 77  SWIT                         PICTURE 9(8)    COMPUTATIONAL.
030040 01  R8                                           COMPUTATIONAL.
030050     02  FILLER                   PICTURE 9(8).
030060     02  RORW                     PICTURE 9(8).
030070     02  LEN                      PICTURE 9(8).
030080 01  MESAGE.
030090     02  ACND                     PICTURE X(10).
030100     02  CNTRS                                    COMPUTATIONAL.
030110      04 A                        PICTURE 9(4).
030120      04 B                        PICTURE 9(4).
030130      04 C                        PICTURE 9(4).
030140      04 D                        PICTURE 9(8).
030150     02  ARRAY                                    COMPUTATIONAL.
030160         04  PTR                  PICTURE 9(8)    OCCURS 115.
040010 01  SCOPE.
040020     02  BIKT.
040030         04  SWIX                 PICTURE X.
040040         04  BUKT                 PICTURE X.
040050         04  INLEN                PICTURE 9(4)    COMPUTATIONAL.
040060     02  SCPBUF.
040070         04  NONE                 PICTURE X(5).
040080         04  FILLER               PICTURE X(475).
040090     02  SCPINP  REDEFINES        SCPBUF.
040100         04  CHAR                 PICTURE X.
040110         04  REST                 PICTURE X(479).
040160     02  SCPOUT  REDEFINES        SCPBUF.
040170         04  SCPLNE               PICTURE X(40)   OCCURS 12.
040120     02  SCPTST  REDEFINES        SCPBUF.
040130         04  VARIABLE.
040140             06  X                PICTURE X       OCCURS 480
040150                                                  DEPENDING L.
050010 01  DIRA.
050020     02  RLHOLD                   PICTURE X(64).
050030     02  CAT-TRAP.
050040         06  CAT-HOLD.
```

```
050050               08  CAT          PICTURE XXX     OCCURS 115.
050060           06  FILLER           PICTURE X(7).
050070           06  CDE-HOLD.
050080               08  CDE          PICTURE XXX     OCCURS 8.
060010 PROCEDURE DIVISION.
060020     ENTER LINKAGE.
060030     ENTRY 'MOD89' USING R8, RTLIST, MESAGE,
060040                          SCOPE, DIRA, SWIT.
060050     ENTER COBOL.
060060 NOP.
060070     EXIT.
060080 BOP.
060142     MOVE SPACE TO SWIX.
060090     IF NONE = 'NONE ' GO TO CONDNS.
060100     IF NONE = 'ALL  ',
060110         MOVE 0 TO PTR (1),
060120                   PTR (2),
060130                   A, B, C,
060140         GO TO CONDNS.
060150 INIT1.
060160     ADD 1 TO I, IF I    A GO TO NXT-VAR.
060170     MOVE CDE (I) TO CAT-NO.
060180     ENTER LINKAGE.
060190     CALL 'IOENQ' USING RDNUM, RTLIST, CAT-INP, CAT-KEY.
060200     ENTER COBOL.
070010     MOVE SPACES TO CAT-NAME (I).
070020     IF ERRWD = 1 MOVE SPACES TO CDE (I) GO TO INIT1.
070030     MOVE CAT-TAG TO CAT-NAME (I).
070040     PERFORM NOP VARYING J FROM 80 BY -1
070050         UNTIL CAT-CHAR (I, J) NOT = ' '
070060         OR J = 0.
070070     MOVE J TO V (I).
070080     GO TO INIT1.
070090 NXT-VAR.
070092     IF INLEN = 0 GO TO NO-MAS.
070100     ADD 1 TO N, IF N    A GO TO NXT-CONST.
```

```
070102      MOVE 80 TO L.
070110      MOVE CAT-NAME (N) TO CONSTANT.
070120      MOVE V (N) TO L.
070130      IF VARIABLE = CONSTANT GO TO DELETE.
070122      GO TO NXT-VAR.
070140 NXT-CONST.
070150      MOVE CHAR TO BUKT.
070160 SHIFT.
070170      MOVE REST TO SCPBUF.
070180      MOVE ' ' TO X (480).
070190 SHOFT.
070200      MOVE BUKT TO X (480).
080000      MOVE 'X' TO SWIX.
080010      ADD 1 TO M,
080020      SUBTRACT 1 FROM INLEN.
080030      MOVE 0 TO N.
080040      GO TO NXT-VAR.
080050 DELETE.
080060      PERFORM DELT1 VARYING I FROM 1 BY 1 UNTIL I   C.
080070      MOVE V (N) TO I.
080080      PERFORM SHIFT I TIMES.
080082      ADD I TO M.
080090      SUBTRACT I FROM INLEN.
080092      MOVE 0 TO N.
080100      GO TO NXT-VAR.
080110 DELT1.
080120      IF CAT (I) = CDE (N) MOVE 0 TO PTR (I).
090010 NO-MAS.
090020      MOVE 0 TO I, MOVE 1 TO J.
090030 CONDNS.
090040 C1. ADD 1 TO I, IF I   C GO TO C3.
090045      IF PTR (I) NOT = 0 ADD 1 TO J GO TO C1.
090050 C2. IF J   C GO TO C1.
090055      IF PTR (J) = 0 ADD 1 TO J, GO TO C2.
090060      MOVE PTR (J) TO PTR (I), MOVE 0 TO PTR (J).
090065      GO TO C1.
```

```
090070 C3. EXIT.
090090 ZEROUT.
090100     IF I   115 NEXT SENTENCE ELSE
090110         MOVE 0 TO PTR (I),
090120         ADD 1 TO I,
090130         GO TO ZEROUT.
090135     COMPUTE C = J - 1.
090140 TSTXT.
090150     IF SWIX = SPACE GO TO PASS.
100010 SLIDE.
100030     PERFORM SHIFT VARYING I FROM M BY 1 UNTIL I = 480.
100040     COMPUTE I = M / 40 + 2.
100050     IF I   12 MOVE  12 TO I.
100060     MOVE 'DID NOT APPEAR IN LIST, TRY AGAIN' TO SCPLNE (I).
100070     COMPUTE LEN = I * 40.
100080     ENTER LINKAGE.
100090     CALL 'GOSYM' USING SCPBUF, LEN.
100100     ENTER COBOL.
100110     MOVE 8 TO RORW.
100120     COMPUTE SWIT = RDNXT + 89 - 1.
100130 RTRN.
100140     ENTER LINKAGE.
100150     RETURN.
100160     ENTER COBOL.
110010 PASS.
110020     MOVE 16 TO RORW.
110030     COMPUTE SWIT = WTNXT + 80 - 1.
110040     GO TO RTRN.
```

```
PROGRAM-ID.  'M91'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  GDSYLENT      PICTURE    9(6)    COMPUTATIONAL.
01  SCOPE.
    02  FILLER     PICTURE   XXXX.
    02  L1       PICTURE   X(40).
   02  FILLER  PICTURE   X(40)   VALUE SPACES.
    02  L2       PICTURE   X(120)  VALUE
    'SOME ERRORS WERE FOUND IN THE ABOVE     ANSWER. PLEASE RE-EN
-   'TER YES-ACCESSION   NO. OR NO IN THE FOLLOWING FORMAT '.
    02  L3      PICTURE    X(120)    VALUE
    'YES-XXXXXXXXXX     OR                        NO'.
01  SW      PICTURE   9   VALUE   ZERO.
LINKAGE SECTION.
01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE    X(4).
    02  RORW    PICTURE    9(6)  COMPUTATIONAL.
    02  LENG    PICTURE    9(6)  COMPUTATIONAL.
01  MESAGE      PICTURE    X(40).
01  DIRA        PICTURE    X.
01  SCOP.
    02  LEGTH      PICTURE      X(4).
    02  L11.
      03   NOL.
           04  NO   PICTURE   XX.
         04   FILLER    PICTURE   X.
        03    YES   REDEFINES   NOL   PICTURE   XXX.
       03    FILLER   PICTURE   X.
        03   ACN    PICTURE     X(10).
    02  FILLER     PICTURE   X(310).
PROCEDURE DIVISION.
    ENTER LINKAGE.
    ENTRY 'MOD91' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
    MOVE  L11   TO  L1.
    IF  LEGTH  GREATER '0039'  GO TO 11.
    IF YES NOT EQUAL TO 'YES'  GO TO 10.
    IF  ACN  EQUAL TO SPACES  MOVE 1 TO SW GO TO MESAGG.
      MOVE ACN TO MESAGE MOVE 1 TO SW GO TO MESAGG.
10. IF NO EQUAL TO 'NO' MOVE 2 TO SW  GO TO MESAGG.
11.  MOVE 33554522  TO SWIT.
ERR.
    MOVE    320   TO  GOSYLENT          LENG.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
AA.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
MESAGG.
    IF  SW EQUAL TO 1  MOVE
    'PLEASE ENTER THE NOTE.'  TO L2  MOVE SPACES TO L3
          MOVE 33554523  TO SWIT  GO TO ERR.
    IF  SW  EQUAL TO  2  MOVE  33554524 TO SWIT
          MOVE 16 TO RORW GO TO AA.
```

```
PROGRAM-ID.  'M92'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  SYMBO-KEY      PICTURE    X(16).
01  ACCESSION-NO      PICTURE    X(16).
01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
01  SCOPE.
    02  FILLER     PICTURE    XXXX.
    02  L1       PICTURE   X(120)  VALUE
    'SOME ERRORS WERE FOUND IN THE ANSWER YOUENTERED. PLEASE RE-E
-   'NTER IT.'.
    02  L2       PICTURE   X(120)     VALUE  SPACES.
    02   L3      PICTURE    X(120)   VALUE  SPACES.
    02  L4      PICTURE    X(40)   VALUE  SPACES.
01  RLDARN      PICTURE     9(6)     COMPUTATIONAL.
01  SW     PICTURE   9   VALUE   ZERO.
LINKAGE SECTION.
01  DIRA.
    02  HIER-COD   PICTURE   X.
    02  FILLER      PICTURE     X(128).
    02  NOTES     PICTURE     X(284).
01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE   X(4).
    02  RORW     PICTURE   9(6)  COMPUTATIONAL.
    02  LENG     PICTURE   9(6)  COMPUTATIONAL.
01  MESAGE.
    02  AC-NO     PICTURE    X(10).
01  SCOP.
    02  LEGTH     PICTURE     X(4).
    02  NOTESS    PICTURE     X(284).
    02   FILLER   PICTURE   X(116).
PROCEDURE DIVISION.
    ENTER LINKAGE.
    ENTRY 'MOD92' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
    IF  LEGTH  GREATER '0284'  GO TO 11.
    MOVE AC-NO TO ACCESSION-NO.
    GO  TO  UPDAT.
11.  MOVE 33554523  TO SWIT.
ERR.
    MOVE    400   TO  GOSYLENT          LENG.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
UPDAT.
    MOVE AC-NO TO SYMBO-KEY.   MOVE 25 TO RLDARN.
    MOVE '1' TO HIER-COD.
    ENTER LINKAGE.
    CALL 'IOENQ' USING RLDARN RTLIST DIRA SYMBO-KEY.
    ENTER COBOL.
    IF HIER-COD EQUAL TO '1'  MOVE 4  TO SW GO TO MESAGG.
    MOVE NOTESS TO NOTES MOVE 25 TO RLDARN.
    MOVE '2' TO HIER-COD.
    ENTER LINKAGE.
    CALL 'IOENQ' USING RLDARN RTLIST DIRA SYMBO-KEY.
    ENTER COBOL.
MESAGG.
    IF SW EQUAL TO 4 MOVE
    'NO RECORD CAN BE FOUND ACCORDING TO THE DATA YOU JUST ENTERE
-   'D. DO YOU WANT TO   UPDATE ANY OTHER NOTE         PLEASE '
        TO  L1   MOVE
    'RE-ENTER YES-ACCESSION NO. OR NO IN THE FOLLOWING FORMAT
```

```
-    '                     YES-XXXXXXXXXX     OR'  TO L2   MOVE
-    'NO'   TO L3   MOVE 33554522  TO SWIT GO TO ERR.
     MOVE
     'THE APPROPRIATE RECORD IS SUCCESSFULLY  UPDATED ACCORDING TO
-    ' THE DATA YOU JUST  ENTERED. DO YOU WANT TO UPDATE ANY OTHER
-    ''   TO L1   MOVE
     'NOTE            PLEASE RE-ENTER          YES-ACCESSION NO. OR
-    ' NO IN THE FOLLOWINGFORMAT '  TO L2  MOVE
     '                                         YES-XXXXXXXXXX     OR
-    '                     NO' TO L3 MOVE 33554522 TO SWIT
          GO TO ERR.
```

```
 PROGRAM-ID.  'M93'.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER.  IBM-360.
 OBJECT-COMPUTER.  IBM-360.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
 01  SCOPE.
     02  FILLER    PICTURE   XXXX.
     02  L1      PICTURE   X(120)  VALUE
     'DO YOU WANT TO UPDATE THE COLLATION     PLEASE ENTER YES-ACC
-    'ESSION NO. OR NO IN THE FOLLOWING FORMAT '.
    02  FILLER  PICTURE   X(40)   VALUE SPACES.
     02  L2  PICTURE  X(120)  VALUE
     'YES-XXXXXXXXXX     OR
-    '                    NO'.
 LINKAGE SECTION.
 01  RTLIST      PICTURE   9(6)  COMPUTATIONAL.
 01  SWIT        PICTURE   9(8)  COMPUTATIONAL.
 01  REG8.
     02  FILLER  PICTURE   X(4).
     02  RORW    PICTURE   9(6)  COMPUTATIONAL.
     02  LENG    PICTURE   9(6)  COMPUTATIONAL.
 01  MESAGE      PICTURE   X.
 01  DIRA        PICTURE   X.
 01  SCOP        PICTURE   X(284).
 PROCEDURE DIVISION.
     ENTER LINKAGE.
     ENTRY 'MOD93' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
     ENTER COBOL.
     MOVE    280   TO  GOSYLENT          LENG.
     ENTER LINKAGE.
     CALL  'GOSYM'  USING  L1  GOSYLENT.
     ENTER COBOL.
     MOVE SCOPE TO SCOP. MOVE  8  TO  RORW. MOVE 33554525 TO SWIT.
```

```
ENTER LINKAGE.
RETURN.
ENTER COBOL.
```

```
PROGRAM-ID.  'M94'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-360.
OBJECT-COMPUTER.  IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  GOSYLENT     PICTURE    9(6)    COMPUTATIONAL.
01  SCOPE.
    02  FILLER    PICTURE   XXXX.
    02  L1      PICTURE   X(40).
   02  FILLER  PICTURE   X(40)   VALUE SPACES.
    02  L2      PICTURE   X(120)  VALUE
    'SOME ERRORS WERE FOUND IN THE ABOVE     ANSWER. PLEASE RE-EN
-   'TER YES-ACCESSION   NO. OR NO IN THE FOLLOWING FORMAT '.
    02  L3     PICTURE    X(120)    VALUE
    'YES-XXXXXXXXXX     OR                  NO'.
01  SW     PICTURE   9   VALUE   ZERO.
LINKAGE SECTION.
01  RTLIST      PICTURE   9(6)  COMPUTATIONAL.
01  SWIT        PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE   X(4).
    02  RORW    PICTURE   9(6)  COMPUTATIONAL.
    02  LENG    PICTURE   9(6)  COMPUTATIONAL.
01  MESAGE      PICTURE   X(40).
01  DIRA        PICTURE   X.
01  SCOP.
    02  LEGTH     PICTURE     X(4).
    02  L11.
      03   NOL.
          04  NO   PICTURE   XX.
        04   FILLER    PICTURE   X.
       03    YES   REDEFINES   NOL   PICTURE   XXX.
      03    FILLER   PICTURE   X.
       03   ACN    PICTURE     X(10).
    02  FILLER      PICTURE    X(310).
PROCEDURE DIVISION.
    ENTER LINKAGE.
    ENTRY 'MOD94' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
    MOVE  L11   TO  L1.
    IF  LEGTH  GREATER '0039'  GO TO 11.
    IF YES NOT EQUAL TO 'YES'  GO TO 10.
    IF  ACN  EQUAL TO SPACES  MOVE 1 TO SW GO TO MESAGG.
     MOVE ACN TO MESAGE MOVE 1 TO SW GO TO MESAGG.
10. IF NO EQUAL TO 'NO' MOVE 2 TO SW  GO TO MESAGG.
11.  MOVE 33554525  TO SWIT.
ERR.
    MOVE    320   TO  GOSYLENT          LENG.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
AA.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
MESAGG.
    IF  SW EQUAL TO 1  MOVE
    'PLEASE ENTER THE COLLATION.'  TO L2  MOVE SPACES TO L3
          MOVE 33554526  TO SWIT  GO TO ERR.
    IF  SW  EQUAL TO  2  MOVE  33554501 TO SWIT
          MOVE 16 TO RORW GO TO AA.
```

```
 PROGRAM-ID.  'M95'.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER.  IBM-360.
 OBJECT-COMPUTER.  IBM-360.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01  SYMBO-KEY      PICTURE    X(16).
 01  ACCESSION-NO      PICTURE    X(16).
 01  GOSYLENT      PICTURE    9(6)    COMPUTATIONAL.
 01  SCOPE.
     02  FILLER     PICTURE    XXXX.
     02  L1       PICTURE   X(120)  VALUE
     'SOME ERRORS WERE FOUND IN THE ANSWER YOUENTERED. PLEASE RE-E
-    'NTER IT.'.
     02  L2       PICTURE   X(120)     VALUE  SPACES.
     02   L3      PICTURE    X(120)   VALUE  SPACES.
     02  L4      PICTURE    X(40)   VALUE  SPACES.
 01  RLDARN      PICTURE     9(6)     COMPUTATIONAL.
 01  SW      PICTURE   9   VALUE   ZERO.
 LINKAGE SECTION.
 01  DIRA.
     02  HIER-COD   PICTURE   X.
     02  FILLER                PICTURE X(449).
     02  COLL      PICTURE      X(40).
 01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
 01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
 01  REG8.
     02  FILLER  PICTURE   X(4).
     02  RORW    PICTURE   9(6)  COMPUTATIONAL.
     02  LENG    PICTURE   9(6)  COMPUTATIONAL.
 01  MESAGE.
     02  AC-NO     PICTURE    X(10).
 01  SCOP.
     02  LEGTH     PICTURE     X(4).
     02  COLLL   PICTURE     X(40).
```

```
    02  FILLER                PICTURE X(436).
PROCEDURE DIVISION.
    ENTER LINKAGE.
    ENTRY 'MOD95' USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
    IF  LEGTH  GREATER '0040'  GO TO 11.
    MOVE AC-NO TO ACCESSION-NO.
    GO  TO  UPDAT.
11.  MOVE 33554526  TO SWIT.
ERR.
    MOVE    400   TO  GOSYLENT          LENG.
    ENTER LINKAGE.
    CALL  'GOSYM'  USING  L1  GOSYLENT.
    ENTER COBOL.
    MOVE SCOPE TO SCOP. MOVE  8  TO  RORW.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
UPDAT.
    MOVE AC-NO TO SYMBO-KEY.   MOVE 25 TO RLDARN.
    MOVE '1' TO HIER-COD.
    ENTER LINKAGE.
    CALL 'IOENQ' USING RLDARN RTLIST DIRA SYMBO-KEY.
    ENTER COBOL.
    IF HIER-COD EQUAL TO '1'  MOVE 4  TO SW GO TO MESAGG.
    MOVE COLLL TO COLL  MOVE  25  TO RLDARN.
    MOVE '2' TO HIER-COD.
    ENTER LINKAGE.
    CALL 'IOENQ' USING RLDARN RTLIST DIRA SYMBO-KEY.
    ENTER COBOL.
MESAGG.
    IF SW EQUAL TO 4 MOVE
    'NO RECORD CAN BE FOUND ACCORDING TO THE DATA YOU JUST ENTERE
-   'D. DO YOU WANT TO   UPDATE ANY OTHER COLLATION    PLEASE '
          TO  L1   MOVE
    'RE-ENTER YES-ACCESSION NO. OR NO IN THE FOLLOWING FORMAT
```

```
-     '                        YES-XXXXXXXXXX      OR'  TO L2  MOVE
-     'NO' TO L3 MOVE 33554525 TO SWIT GO TO ERR.
      MOVE
      'THE APPROPRIATE RECORD IS SUCCESSFULLY  UPDATED ACCORDING TO
-     ' THE DATA YOU JUST  ENTERED. DO YOU WANT TO UPDATE ANY OTHER
-     ''   TO L1   MOVE
      'COLLATION       PLEASE RE-ENTER            YES-ACCESSION NO. OR
-     ' NO IN THE FOLLOWINGFORMAT '  TO L2  MOVE
      '                                           YES-XXXXXXXXXX     OR
-     '                        NO' TO L3 MOVE 33554525 TO SWIT
          GO TO ERR.
```

```
IDENTIFICATION DIVISION.
PROGRAM-ID. 'M99'.
AUTHOR. 'E.S.DAHL'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-360.
OBJECT-COMPUTER. IBM-360.
DATA DIVISION.
LINKAGE SECTION.
77  A          PICTURE X.
77  B          PICTURE X.
77  C          PICTURE X.
77  D          PICTURE X.
77  E          PICTURE X.
01  R8                             COMPUTATIONAL.
    02  FILLER     PICTURE 9(8).
    02  TPW        PICTURE 9(8).
PROCEDURE DIVISION.
    ENTER LINKAGE.
    ENTRY 'MOD99' USING R8, A B C D E.
    ENTER COBOL.
    MOVE 1 TO TPW.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
```

```
C     DNTREE SUBROUTINE                                                 P010L010
C     S S MILLWRIGHT MAY 22, 1968                                       P010L020
C      ED S DAHL 11 JUL 68                                              P010L022
C                                                                       P010L030
C     THIS SUBROUTINE DETERMINES THE NEXT LOWER LEVEL (MORE SPECIFIC)   P010L040
C     CATEGORIES TO WHICH A GIVEN SUBJECT BELONGS                       P010L050
C                                                                       P010L060
C                                                                       P010L070
      SUBROUTINE DNTREE (MNOS,MK,IORTN,MBUF,MSCP)                       P010L080
      DIMENSION NOS(4,10),NUMBR(6),K(7),IBUF(23),H(2),ISCP(120),IMSG(22)P010L090
     2,         MNOS(4,10),MK(7),MBUF(23),MSCP(120),IORTN(1)            P010L092
      INTEGER TEMP2(17),TEMP3                                           P010L100
      INTEGER*2 H                                                       P010L102
      EQUIVALENCE (K(1),KOUNT),(K(2),J),(K(6),H(1)),(K(7),KODE),        P010L103
     2             (IBUF(1),LOST),(IBUF(2),TEMP2(1)),(IBUF(23),TEMP3),  P010L104
     3             (NUMBR(5),MORE),(NUMBR(6),KK)                        P010L105
      DATA IMSG/'NEXT MORE SPECIFIC CATEGORIES ARE   NOT ON FILE SUBJECTP010L106
     2 BELONGS TO MOST SPECIFIC LEVEL  '/                               P010L107
C.....PASS VALUES IN.....                                               P010L110
      DO 301 M=1,4                                                      P010L111
      DO 301 N=1,10                                                     P010L112
  301 NOS(M,N)=MNOS(M,N)                                                P010L113
      DO 302 M=1,7                                                      P010L114
  302 K(M)=MK(M)                                                        P010L115
      DO 303 M=1,23                                                     P010L116
  303 IBUF(M)=MBUF(M)                                                   P010L117
      DO 304 M=1,120                                                    P010L118
  304 ISCP(M)=MSCP(M)                                                   P010L119
      DO 320 M=1,6
  320 NUMBR(M)=MK(M-7)
      IHIT=0                                                            P010L119
      IF(KOUNT.EQ.0)GOTO 310                                            P010L120
      IF(J.NE.0)GOTO 106                                                P010L121
C.....WRITE.....NEXT MORE SPECIFIC CATEGORIES ARE.....                  P010L121
      N=H(1)*10+1                                                       P010L123
      DO 201 I=1,9                                                      P010L124
      ISCP(N)=IMSG(I)                                                   P010L125
  201 N=N+1                                                             P010L126
      H(1)=H(1)+1                                                       P010L122
C.....SCAN NOS MATRIX FOR MATCHING KODES.....                           P010L127
      IF(J.EQ.0)J=1                                                     P010L130
  101 IF(NOS(1,J).EQ.KODE)GO TO 102                                     P010L140
  103 J=J+1                                                             P010L150
      IF(J.GT.KOUNT)GO TO 312                                           P010L160
      GO TO 101                                                         P010L170
C.....A MATCH HAS BEEN FOUND.....                                       P010L180
  102 N=14                                                              P010L190
      IHIT=1                                                            P010L191
      CALL IOENQ (N,IORTN,IBUF,NOS(1,J))                                P010L195
C.....ARE THERE ANY SUB-LEVELS FOR THIS SUBJECT.....                    P010L200
      IF(TEMP3.EQ.0)GO TO 104                                           P010L210
C.....THERE ARE SUB-LEVELS.....                                         P020L010
  105 N=15                                                              P020L120
      IF(H(1).GT.9)GOTO 309                                             P020L14
      H(2)=90                                                           P020L120
      CALL IOENQ (N,IORTN,NUMBR,TEMP3)                                  P020L125
      KK=TEMP3
      N=14                                                              P020L130
      CALL IOENQ (N,IORTN,IBUF,NUMBR)                                   P020L135
      IF(IBUF(1).EQ.0)IHIT=1
C.....WRITE.....OUTPUT DATA.....                                        P020L140
      N=67                                                              P020L142
      CALL SCPTXT (IBUF(2),N,ISCP,H(1))                                 P020L144
  106 TEMP3=KK
C.....ARE THERE MORE SUB-LEVELS.....                                    P020L050
      IF(MORE.EQ.1)GO TO 105                                            P020L060
      GO TO 103                                                         P020L070
C.....THERE ARE NO SUB-LEVELS.....                                      P020L080
C.....WRITE.....SUBJECT BELONGS TO MOST SPECIFIC LEVEL.....             P020L190
  104 N=H(1)*10+1                                                       P020L192
      DO 202 I=13,22                                                    P020L193
      ISCP(N)=IMSG(I)                                                   P020L194
  202 N=N+1                                                             P020L195
      IF(H(2).NE.90)H(2)=98                                             P020L195
      GO TO 103                                                         P020L100
C.....PASS VALUES OUT.....                                              P040L010
  312 IF(H(2).EQ.98)GOTO 305                                            P030L010
      IF(IHIT.EQ.1)H(2)=99                                              P030L010
  305 IF(IHIT-0)309,300,309                                             P030L030
  300 N=H(1)*10+1                                                       P030L040
      DO 310 I=10,12                                                    P030L050
      ISCP(N)=IMSG(I)                                                   P030L060
  310 N=N+1                                                             P030L070
      H(2)=98                                                           P030L080
      H(1)=H(1)+1                                                       P030L090
  309 DO 306 M=1,7                                                      P030L100
  306 MK(M)=K(M)                                                        P040L012
      DO 307 M=1,23                                                     P040L013
  307 MBUF(M)=IBUF(M)                                                   P040L014
      DO 308 M=1,120                                                    P040L015
  308 MSCP(M)=ISCP(M)                                                   P040L016
      DO 321 M=1,6
  321 MK(M-7)=NUMBR(M)
      RETURN                                                            P040L020
      END                                                               P030L070
C     UPTREE SUBROUTINE                                                 P010L010
C     S S MILLWRIGHT MAY 21, 1968                                       P010L020
C     ED S DAHL 9 JUL 68                                                P010L022
C                                                                       P010L030
C     THIS SUBROUTINE DETERMINES THE NEXT HIGHER LEVEL                  P010L040
C     (MORE GENERAL) CATEGORY TO WHICH A GIVEN SUBJECT BELONGS          P010L050
C                                                                       P010L060
C                                                                       P010L070
      SUBROUTINE UPTREE (MNOS,MK,IORTN,MBUF,MSCP)                       P010L080
      DIMENSION NOS(4,10),KHAR(3),K(7),IBUF(23),H(2),ISCP(120),IMSG(24),P010L090
     2          MNOS(4,10),MK(7),MBUF(23),MSCP(120),IORTN(1)            P010L091
      INTEGER HIT,TEMP1(4),TEMP2(17),TEMP3                              P010L100
      INTEGER*2 H                                                       P010L101
      EQUIVALENCE (K(1),KOUNT),(K(2),J),(K(6),H(1)),(K(7),KODE),          P010L102
     2            (IBUF(1),LOST),(IBUF(2),TEMP2(1)),(IBUF(23),TEMP3)      P010L103
      DATA KHAR/'000.00.00    '/                                          P010L110
      DATA IMSG/'NEXT MORE GENERAL CATEGORY IS    OR SUBJECT BELONGS TO P010L111
     2MOST GENERAL LEVEL   NOT ON FILE '/                                 P010L112
C.....PASS VALUES IN.....                                                  P010L113
      DO 301 M=1,4                                                        P010L114
      DO 301 N=1,10                                                       P010L115
  301 NOS(M,N)=MNOS(M,N)                                                  P010L116
      DO 302 M=1,7                                                        P010L117
  302 K(M)=MK(M)                                                          P010L118
      DO 303 M=1,23                                                       P010L119
  303 IBUF(M)=MBUF(M)                                                     P010L120
      DO 304 M=1,120                                                      P010L121
  304 ISCP(M)=MSCP(M)                                                     P010L122
      N=H(1)*10+1                                                         P010L120
      DO 201 I=1,8                                                        P010L121
      ISCP(N)=IMSG(I)                                                     P010L122
  201 N=N+1                                                               P010L123
      H(1)=H(1)+1                                                         P010L124
      IF (KOUNT.EQ.0)GOTO 305                                             P010L125
C.....SCAN NOS MATRIX FOR MATCHING KODES.....                              P010L130
      HIT=0                                                               P010L140
      J=1                                                                 P010L150
  101 IF(NOS(1,J).EQ.KODE)GO TO 102                                       P010L160
  103 J=J+1                                                               P010L170
      IF(J.GT.KOUNT)GOTO 305                                              P010L180
      GO TO 101                                                           P010L190
C.....PASS VALUES OUT.....                                                 P010L191
  305 IF (HIT-0)309,300,309                                               P010L191
  300 H(2)=98                                                             P010L191
      N=H(1)*10+1                                                         P010L191
      DO 310 I=21,24                                                      P010L191
      ISCP(N)=IMSG(I)                                                     P010L191
  310 N=N+1                                                               P010L191
      H(1)=H(1)+1                                                         P010L191
  309 DO 306 M=1,7                                                   P010L191
  306 MK(M)=K(M)                                                     P010L193
      DO 307 M=1,23                                                  P010L194
  307 MBUF(M)=IBUF(M)                                                P010L195
      DO 308 M=1,120                                                 P010L196
  308 MSCP(M)=ISCP(M)                                                P010L197
      RETURN                                                         P010L198
C.....A MATCH HAS BEEN FOUND.....                                     P010L200
  102 IF(HIT.EQ.0)GO TO 104                                          P010L210
      N=H(1)*10+1                                                    P010L220
      ISCP(N)=IMSG(9)                                                P010L221
      H(1)=H(1)+1                                                    P010L222
      GO TO 105                                                      P010L230
  104 HIT=1                                                          P020L010
C.....DETERMINE THE FORM OF THE CATEGORY NUMBER.....                  P020L020
  105 CALL WTFORM(NOS(1,J),NFORM)                                    P020L030
      KLAX=0
      IF(NFORM-2)106,107,108                                         P020L040
C.....NFORM=1, NO MORE GENERAL CATEGORY EXISTS.....                    P020L050
  106 N=H(1)*10+1                                                    P020L060
      N=N-10
      DO 203 I=10,19                                                 P020L061
      ISCP(N)=IMSG(I)                                                P020L062
  203 N=N+1                                                          P020L063
      H(1)=H(1)+1                                                    P020L064
      H(2)=98                                                        P020L064
      GO TO 103                                                      P020L070
C.....NFORM=2, CHANGE FROM ANN.NNN.00.00 TO ANN.000.00.00.....         P020L080
  107 TEMP1(4)=KHAR(3)                                               P020L090
      TEMP1(3)=KHAR(2)                                               P020L100
      TEMP1(2)=KHAR(1)                                               P020L110
  109 TEMP1(1)=NOS(1,J)                                              P020L120
      GO TO 110                                                      P020L130
  108 IF(NFORM.EQ.4)GO TO 111                                        P020L140
C.....NFORM=3, CHANGE FROM ANN.NNN.NN.00 TO ANN.NNN.00.00.....         P020L150
      TEMP1(4)=KHAR(3)                                               P020L160
      TEMP1(3)=KHAR(2)                                                  P020L170
  112 TEMP1(2)=NOS(2,J)                                                 P020L180
      GO TO 109                                                         P020L190
C.....NFORM=4, CHANGE FROM ANN.NNN.NN.NN TO ANN.NNN.NN.00.....          P020L200
  111 TEMP1(4)=KHAR(3)                                                  P020L210
      TEMP1(3)=NOS(3,J)/256*256-16                                      P020L220
      GO TO 112                                                         P020L230
C.....READ THE NUMERIC FILE TO RETRIEVE THE NEXT MORE GENERAL CATEGORY..P030L010
  110 L=14                                                              P030L020
      CALL IOENQ (L,IORTN,IBUF,TEMP1)                                   P030L021
C.....IS THIS RECORD ON DISK.....                                       P030L030
      IF(LOST.EQ.1)GO TO 113                                            P030L040
C.....YEP.....                                                          P030L050
      N=67                                                              P030L060
      CALL SCPTXT (IBUF(2),N,ISCP,H(1))                                 P030L061
      GO TO 103                                                         P030L070
C.....NOPE.....                                                         P030L080
  113 N=H(1)*10+1                                                       P030L090
      DO 204 I=21,24                                                    P030L091
      ISCP(N)=IMSG(I)                                                   P030L092
  204 N=N+1                                                             P030L093
      H(1)=H(1)+1                                                       P030L094
      GO TO 103                                                         P030L100
      END                                                               P040L090
C     WHAT-FORM SUBROUTINE                                              P01LN010
C     S S MILLWRIGHT JANUARY 19, 1968                                   P01LN020
C     DETERMINES 'FORM' OF THE CATEGORY NUMBER NUM(I)                   P01LN030
C     IF                                                                P01LN040
C     ANN.000.00.00 NFORM=1                                             P01LN050
C     ANN.NNN.00.00 NFORM=2                                             P01LN060
C     ANN.NNN.NN.00 NFORM=3                                             P01LN070
C     ANN.NNN.NN.NN NFORM=4                                             P01LN080
C                                                                       P01LN090
C                                                                       P01LN100
      SUBROUTINE WTFORM(NUM,NFORM)                                      P01LN110
      DIMENSION NUM(4),KHAR(3)                                          P01LN120
      DATA KHAR(1)/'0   '/                                              P01LN130
      DATA KHAR(2)/'00.0'/                                              P01LN140
      DATA KHAR(3)/'000.'/                                              P01LN150
      IF(NUM(4).NE.KHAR(1))GO TO 101                                    P01LN160
      IF(NUM(3).NE.KHAR(2))GO TO 102                                    P01LN170
      IF(NUM(2)-KHAR(3))103,104,103                                     P01LN180
  102 NTEST=NUM(3)/256*256-16                                           P01LN190
      IF(NTEST-NUM(3))101,108,101                                       P01LN200
  101 NFORM=4                                                           P02LN030
  109 RETURN                                                            P02LN040
  103 NFORM=2                                                           P02LN050
      GO TO 109                                                         P02LN060
  104 NFORM=1                                                           P02LN070
      GO TO 109                                                         P02LN080
  108 NFORM=3                                                           P02LN090
      GO TO 109                                                         P02LN100
      END                                                               P02LN110
```

```
010010 IDENTIFICATION DIVISION.
010020 PROGRAM-ID. 'ROCALP'.
010030 AUTHOR. S S MILLWRIGHT.
010040 DATE-WRITTEN. MAY 2, 1968.
010050 REMARKS. READS OPENS AND CLOSES ALPHA CATEGORY FILE.
010060 ENVIRONMENT DIVISION.
010070 CONFIGURATION SECTION.
010080 SOURCE-COMPUTER. IBM-360.
010090 OBJECT-COMPUTER. IBM-360.
010100 INPUT-OUTPUT SECTION.
010110 FILE-CONTROL.
010120     SELECT ALPHA-FILE ASSIGN TO 'SYS001' DIRECT-ACCESS 2311
010130     ORGANIZATION IS INDEXED ACCESS IS RANDOM
010140     RECORD KEY IS CAT-NAME
010150     SYMBOLIC KEY IS CAT-NAME-KEY.
010160 DATA DIVISION.
010170 FILE SECTION.
010180 FD  ALPHA-FILE
010190     LABEL RECORDS ARE STANDARD
010200     RECORDING MODE IS F
010210     DATA RECORD IS ALPHA-RECD.
020010 01  ALPHA-RECD.
020020     02 FILLER                PICTURE X(1).
020030     02 CAT-NAME              PICTURE X(67).
020040     02 KOUNT                 PICTURE 9(3).
020050     02 CAT-NUMB              PICTURE X(13) OCCURS 10 TIMES.
020060 WORKING-STORAGE SECTION.
020070 77  TEMP                     PICTURE 9(3).
020080 77  CAT-NAME-KEY             PICTURE X(67).
030010 LINKAGE SECTION.
030020 01  CAT-CALL                 PICTURE X(67).
030030 01  USR-BUFR.
030032     02  ERRWD          PICTURE 9(8)    COMPUTATIONAL.
030040     02  CAT-FND        PICTURE X(68).
030050     02  TALLI          PICTURE 9999.
030060     02  TBL            PICTURE X(16)   OCCURS 10.
```

```
030070 PROCEDURE DIVISION.
030080     ENTER LINKAGE.
030090     ENTRY 'RDALP' USING USR-BUFR, CAT-CALL.
030110     ENTER COBOL.
030112     MOVE CAT-CALL TO CAT-NAME-KEY.
030115     MOVE CAT-NAME-KEY TO CAT-NAME.
030120     READ ALPHA-FILE INVALID KEY GO TO OOPS.
030130     MOVE KOUNT TO TALLI.
030132     MOVE CAT-NAME TO CAT-FND.
030140     MOVE 1 TO TEMP.
030150 LOOP-THE-LOOP.
030160     MOVE CAT-NUMB (TEMP) TO TBL (TEMP).
030170     ADD 1 TO TEMP.
030180     IF TEMP IS NOT GREATER THAN KOUNT GO TO LOOP-THE-LOOP.
030190     MOVE ZERO TO ERRWD.
030200 BRANCH-BACK.
030210     ENTER LINKAGE.
030220     RETURN.
030230     ENTER COBOL.
040010 OOPS.
040020     MOVE 1 TO ERRWD.
040030     GO TO BRANCH-BACK.
040040     ENTER LINKAGE.
040050     ENTRY 'OPEALP'.
040060     ENTER COBOL.
040070     OPEN INPUT ALPHA-FILE.
040080     GO TO BRANCH-BACK.
040090     ENTER LINKAGE.
040100     ENTRY 'CLOALP'.
040110     ENTER COBOL.
040120     CLOSE ALPHA-FILE.
040130     GO TO BRANCH-BACK.
```

```
010010 IDENTIFICATION DIVISION.
010020 PROGRAM-ID. 'ROCNUM'.
010030 AUTHOR. S S MILLWRIGHT.
010040 DATE-WRITTEN. MAY 13, 1968.
010050 REMARKS. READS OPENS AND CLOSES NUMERIC CATEGORY FILE.
010060 ENVIRONMENT DIVISION.
010070 CONFIGURATION SECTION.
010080 SOURCE-COMPUTER. IBM-360.
010090 OBJECT-COMPUTER. IBM-360.
010100 INPUT-OUTPUT SECTION.
010110 FILE-CONTROL.
010120     SELECT NUMBR-FILE ASSIGN TO 'SYS002' DIRECT-ACCESS 2311
010130     ORGANIZATION IS INDEXED ACCESS IS RANDOM
010140     RECORD KEY IS CAT-NUMB
010150     SYMBOLIC KEY IS CAT-NUMB-KEY.
010160 DATA DIVISION.
010170 FILE SECTION.
010180 FD  NUMBR-FILE
010190     LABEL RECORDS ARE STANDARD
010200     RECORDING MODE IS F
010210     DATA RECORD IS NUMBR-RECD.
020010 01  NUMBR-RECD.
020020        02 FILLER                  PICTURE X(1).
020030        02 CAT-NAME                PICTURE X(67).
020040        02 CAT-NUMB                PICTURE X(13).
020050        02 OTHER-FILE-ADDR         PICTURE 9(5).
020060 WORKING-STORAGE SECTION.
020070 77  CAT-NUMB-KEY                  PICTURE X(13).
030010 LINKAGE SECTION.
030020 01  CAT-CALL                      PICTURE X(13).
030030 01  USR-BUFR.
030032     02  ERRWD             PICTURE 9(8)    COMPUTATIONAL.
030040     02  NAME-FND          PICTURE X(68).
030042     02  CAT-NO            PICTURE X(16).
030050     02  FILEF-NO          PICTURE 9(8)    COMPUTATIONAL.
030060 PROCEDURE DIVISION.
```

```
030070     ENTER LINKAGE.
030080     ENTRY 'RDNUM' USING USR-BUFR, CAT-CALL.
030100     ENTER COBOL.
030102     MOVE CAT-CALL TO CAT-NUMB-KEY.
030105     MOVE CAT-NUMB-KEY TO CAT-NUMB.
030110     READ NUMBR-FILE INVALID KEY GO TO SORRY-ABOUT-THAT.
030120     MOVE ZERO TO ERRWD.
030130     MOVE CAT-NAME TO NAME-FND.
030140     MOVE OTHER-FILE-ADDR TO FILEF-NO.
030150 YANKEE-GO-HOME.
030160     ENTER LINKAGE.
030170     RETURN.
030180     ENTER COBOL.
030190 SORRY-ABOUT-THAT.
030200     MOVE 1 TO ERRWD.
030210     GO TO YANKEE-GO-HOME.
040010     ENTER LINKAGE.
040020     ENTRY 'OPENUM'.
040030     ENTER COBOL.
040040     OPEN INPUT NUMBR-FILE.
040050     GO TO YANKEE-GO-HOME.
040060     ENTER LINKAGE.
040070     ENTRY 'CLONUM'.
040080     ENTER COBOL.
040090     CLOSE NUMBR-FILE.
040100     GO TO YANKEE-GO-HOME.
```

```
      SUBROUTINE RWACN1(BUF,KEY)
      INTEGER BUF
      DIMENSION BUF(10)
      DEFINE FILE 2(999,52,E,IREC)
      IREC=1
      READ(2'IREC,100) (BUF(I),I=1,10)
  100 FORMAT(4A4,6I6)
      IREC=IREC-1
      GO TO(1,2,3,4,5,6),KEY
    1 KEY=BUF(5)
      BUF(5)=BUF(5)+1
      GO TO 7
    2 KEY=BUF(6)
      BUF(6)=BUF(6)+1
      GO TO 7
    3 KEY=BUF(7)
      BUF(7)=BUF(7)+1
      GO TO 7
    4 KEY=BUF(8)
      BUF(8)=BUF(8)+1
      GO TO 7
    5 KEY=BUF(9)
      BUF(9)=BUF(9)+1
      GO TO 7
    6 KEY=BUF(10)
      BUF(10)=BUF(10)+1
    7 IREC=1
      WRITE(2'IREC,100) (BUF(I),I=1,10)
      RETURN
      END
C     READ FILEF SUBROUTINE                                             P010L010
C     S S MILLWRIGHT MAY 14, 1968                                       P010L020
C     ED S DAHL 2 JUL 68
C                                                                       P010L030
C                                                                       P010L040
      SUBROUTINE RFILEF(NUMBR,NOREC)                                    P010L050
      DIMENSION NUMBR(5)                                                P010L060
      DEFINE FILE 1(13000,14,E,NOREC)                                   P010L070
      READ(1'NOREC,1)(NUMBR(I),I=1,5)                                   P010L080
      RETURN                                                            P010L090
    1 FORMAT(3A4,A1,I1)                                                 P010L100
      END                                                               P010L110
```

6

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'CLRSCX'.
001030 AUTHOR. 'E.S.DAHL'.
001040 ENVIRONMENT DIVISION.
001050 CONFIGURATION SECTION.
001060 SOURCE-COMPUTER. IBM-360.
001070 OBJECT-COMPUTER. IBM-360.
001080 DATA DIVISION.
001090 WORKING-STORAGE SECTION.
001100 77  RDSCRT                PICTURE 9(8)    COMPUTATIONAL VALUE 16.
001110 LINKAGE SECTION.
001120 77  RDLIST                PICTURE 9(8)    COMPUTATIONAL.
001130 01  MESAGE.
001140     02  FILLER            PICTURE X(10).
001150     02  COMP                              COMPUTATIONAL.
001160         04  I             PICTURE 9(4).
001180         04  K             PICTURE 9(8).
001170         04  J             PICTURE 9(8).
001190         04  ARRAY.
001200             06  L         PICTURE 9(8)    OCCURS 114.
001210         04  LINK.
001220             06  M         PICTURE 9(4).
001230             06  N         PICTURE 9(4).
002010 PROCEDURE DIVISION.
002020     ENTER LINKAGE.
002030     ENTRY 'CLRSCRX' USING RTLIST, MESAGE.
002040     ENTER COBOL.
002050     IF K ) 115 GO TO B.
002060 A.  ENTER LINKAGE.
002070     CALL 'IOENQ' USING RDSCRT, RTLIST, ARRAY, LINK.
002080     ENTER COBOL.
002090     IF N NOT = 0 GO TO A.
002100 B.  MOVE 0 TO I, J, K.
002110     ENTER LINKAGE.
002120     RETURN.
-02130     ENTER COBOL.
```

```
001000 IDENTIFICATION DIVISION.
001010 PROGRAM-ID. 'SCPTEXT'.
001020 AUTHOR. 'E.S.DAHL'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 01  COUNTERS                                COMPUTATIONAL.
001100     02  CNTRA               PICTURE 9(4).
001110     02  CNTRB               PICTURE 9(4).
001120     02  CNTRC               PICTURE 9(4).
001130     02  CNTRD               PICTURE 9(4).
001140 LINKAGE SECTION.
001150 77  LINDX                   PICTURE 9(4)    COMPUTATIONAL.
001160 77  LENGTH                  PICTURE 9(8)    COMPUTATIONAL.
001170 01  MESSAGE.
001180     02  TEXT                PICTURE X       OCCURS 480.
001190 01  BUFFER.
001200     02  SCOOP                               OCCURS 12.
001210         04  SCOPE           PICTURE X       OCCURS 40.
002010 PROCEDURE DIVISION.
002020     ENTER LINKAGE.
002030     ENTRY 'SCPTXT' USING MESSAGE, LENGTH, BUFFER, LINDX.
002040     ENTER COBOL.
002050 INITL.
002060     COMPUTE CNTRA = LENGTH.
002070     MOVE 1 TO CNTRB.
002080 SCAN-EOD.
002090     PERFORM NO-OP VARYING CNTRA FROM CNTRA BY -1
002100             UNTIL TEXT (CNTRA) NOT EQUAL TO SPACE
002110             OR CNTRA = ZERO.
002120 NO-OP. EXIT.
002130 1.          IF CNTRA = 0,
002140                 ADD 1 TO LINDX,
002150                 GO TO ENDE.
002160             IF CNTRA ) 41,
002170                 MOVE CNTRA TO CNTRC,
002180                 GO TO LOAD-LINE.
002190 SCAN-EOL.
002200 2.  COMPUTE CNTRC = CNTRB + 40.
002210 3.  PERFORM NO-OP VARYING CNTRC FROM CNTRC BY -1
002220             UNTIL TEXT (CNTRC) = SPACE
002230             OR CNTRC ) CNTRB.
003010             IF CNTRC ) CNTRB,
003020                 PERFORM 2.
003030 LOAD-LINE.
003040     IF LINDX ) 12 ADD 1 TO LINDX.
003050     PERFORM MOVE-TEXT VARYING CNTRD FROM 1 BY 1
003060             UNTIL CNTRD   CNTRC.
003070     IF CNTRB ) CNTRA GC TO SCAN-EOL.
003080 ENDE.
003090     MOVE CNTRB TO LENGTH.
003100     ENTER LINKAGE.
003110     RETURN.
003120     ENTER COBOL.
003130 MOVE-TEXT.
003140     MOVE TEXT (CNTRB) TO SCOPE (LINDX, CNTRD),
003150     ADD 1 TO CNTRB.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'SUBUPD'.
001030 AUTHOR. 'WAYNE JONES'.
001040 ENVIRONMENT DIVISION.
001050 CONFIGURATION SECTION.
001060 SOURCE-COMPUTER. IBM-360.
001070 OBJECT-COMPUTER. IBM-360.
001080 INPUT-OUTPUT SECTION.
001090 FILE-CONTROL.
              SELECT SUBJECT-IN ASSIGN TO 'SYS010' UTILITY 2311.
001110 DATA DIVISION.
001111 FILE SECTION.
001120 FD   SUBJECT-IN
001130      LABEL RECORDS ARE STANDARD
001140      RECORDING MODE IS F
001150      BLOCK 20 RECORDS
001160      DATA RECORD IS SUB-IN-REC.
001170 01   SUB-IN-REC.
001180      02  ACN-IN                PICTURE 9(8) COMPUTATIONAL.
001190      02  SUBJ-IN               PICTURE X(68).
002010 WORKING-STORAGE SECTION.
002020 77   FORT-KEY                  PICTURE 9(8) COMPUTATIONAL.
002030 77   SUBJ1-PT-HOLD             PICTURE 9(8) COMPUTATIONAL.
002040 77   SUB-TEST                  PICTURE X(80).
002050 77   REWR-SUB2                 PICTURE 9(6) COMPUTATIONAL.
002060 77   NEW-SUBJ                  PICTURE 9(6) COMPUTATIONAL.
003010 01   DIRA1.
003020      02  FILLER                PICTURE X(16).
003030      02  NEXT-AV-AUT           PICTURE 9(6) COMPUTATIONAL.
003040      02  NEXT-AV-T1            PICTURE 9(6) COMPUTATIONAL.
003050      02  NEXT-AV-T2            PICTURE 9(6) COMPUTATIONAL.
003060      02  NEXT-AV-SU1           PICTURE 9(6) COMPUTATIONAL.
003070      02  NEXT-AV-ACN           PICTURE 9(6) COMPUTATIONAL.
003080      02  NEXT-AV-SU2           PICTURE 9(6) COMPUTATIONAL.
003090 01   DIRA2.
003100      02  FILLER                PICTURE X(16).
```

```
003110      02  N-T-T1                  PICTURE 9(6) COMPUTATIONAL.
003120      02  INDEX-T-SUB             PICTURE 9(6) COMPUTATIONAL.
003130      02  N-T-AUT                 PICTURE 9(6) COMPUTATIONAL.
003140      02  OTHER-AUTH              PICTURE 9(6) COMPUTATIONAL.
003150      02  A-L                     PICTURE 9(6) COMPUTATIONAL.
003160      02  N-N-ACN                 PICTURE 9(6) COMPUTATIONAL.
003170 01  DIRA3.
003180      02  SUBJECT1                PICTURE X(80).
003190      02  FRST-REC2               PICTURE 9(6) COMPUTATIONAL.
003200      02  MESH                    PICTURE X(10).
003210 01  DIRA4.
003220      02  NDEX-ACCN               PICTURE 9(6) COMPUTATIONAL.
003230      02  NEXT-REC2               PICTURE 9(6) COMPUTATIONAL.
003240      02  OTHER-SUBJ              PICTURE 9(6) COMPUTATIONAL.
003250      02  PAGE-NO                 PICTURE 9(6) COMPUTATIONAL.
004010 01  DIRA5.
004020      02  ERR-IND1                PICTURE X.
004030      02  SUBJ-ISAM               PICTURE X(80).
004040      02  NDEX-TO-SUB1            PICTURE 9(6).
004050 01  DIRA6.
004060      02  AUT-HOR                 PICTURE X(213).
004070      02  FILLER                  PICTURE 9(6) COMPUTATIONAL.
004080      02  INDEX-T-ACCN            PICTURE 9(6).
004090      02  FILLER                  PICTURE 9(6) COMPUTATIONAL.
004100      02  FILLER                  PICTURE X(8).
005010 PROCEDURE DIVISION.
005030     OPEN INPUT SUBJECT-IN.
005031     ENTER LINKAGE.
005032     CALL 'OPNRN'.
005033     ENTER COBOL.
005040 START.
005050     MOVE 1 TO FORT-KEY.
005060     ENTER LINKAGE.
005070     CALL 'ACNFOR' USING DIRA1 FORT-KEY.
005080     ENTER COBOL.
005090 A1.
```

```
005100      READ SUBJECT-IN AT END GO TO A6.
005110      MOVE ACN-IN TO FORT-KEY.
005120 A7.
005130      ENTER LINKAGE.
005140      CALL 'ACNFOR' USING DIRA2 FORT-KEY.
005150      ENTER COBOL.
005160      IF INDEX-T-SUB = 0 GO TO D.
005170      MOVE INDEX-T-SUB TO FORT-KEY SUBJ1-PT-HOLD.
005180 A2.
005190      ENTER LINKAGE.
005200      CALL 'SU1FOR' USING DIRA3 FORT-KEY.
005210      ENTER COBOL.
005220      MOVE SUBJ-IN TO SUB-TEST.
005230      IF SUB-TEST = SUBJECT1 GO TO A1.
005240      MOVE FRST-REC2 TO FORT-KEY.
005250 A3.
006010      ENTER LINKAGE.
006020      CALL 'SU2FOR' USING DIRA4 FORT-KEY.
006030      ENTER COBOL.
006040      IF NDEX-ACCN IS NOT EQUAL TO ACN-IN GO TO A4.
006050      IF OTHER-SUBJ = SUBJ1-PT-HOLD GO TO A5.
006060      MOVE OTHER-SUBJ TO FORT-KEY.
006070      GO TO A2.
006080 A4.
006090      MOVE NEXT-REC2 TO FORT-KEY.
006100      GO TO A3.
006110 A5.
006120      MOVE FORT-KEY TO REWR-SUB2.
006130      SUBTRACT 1 FROM REWR-SUB2.
006140 B.
006150      ENTER LINKAGE.
006160      CALL 'SEQSUBIO' USING DIRA5 SUB-TEST.
006170      ENTER COBOL.
006180      IF ERR-IND1 = '1' GO TO C.
006190      MOVE NDEX-TO-SUB1 TO FORT-KEY NEW-SUBJ.
006200      ENTER LINKAGE.
```

```
006210     CALL 'RWRSUB' USING DIRA5 SUB-TEST.
006220     ENTER COBOL.
006230     ENTER LINKAGE.
006240     CALL 'SU1FOR' USING DIRA3 FORT-KEY.
006250     ENTER COBOL.
007010     MOVE FRST-REC2 TO FORT-KEY.
007020 B1.
007030     ENTER LINKAGE.
007040     CALL 'SU2FOR' USING DIRA4 FORT-KEY.
007050     ENTER COBOL.
007060     IF NEXT-REC2 IS NOT EQUAL TO 0 GO TO B2.
007070     SUBTRACT 1 FROM FORT-KEY.
007080     MOVE NEXT-AV-SU2 TO NEXT-REC2.
007090     ENTER LINKAGE.
007100     CALL 'WTSUB2' USING DIRA4 FORT-KEY.
007110     ENTER COBOL.
007120 B3.
007130     MOVE NEXT-AV-SU2 TO FORT-KEY.
007140     MOVE ACN-IN TO NDEX-ACCN.
007150     MOVE SUBJ1-PT-HOLD TO OTHER-SUBJ.
007160     MOVE 0 TO NEXT-REC2 PAGE-NO.
007170     ENTER LINKAGE.
007180     CALL 'WTSUB2' USING DIRA4 FORT-KEY.
007190     ENTER COBOL.
007200     MOVE REWR-SUB2 TO FORT-KEY.
007210     ENTER LINKAGE.
007220     CALL 'SU2FOR' USING DIRA4 FORT-KEY.
007230     ENTER COBOL.
007240     SUBTRACT 1 FROM FORT-KEY.
007250     MOVE NEW-SUBJ TO OTHER-SUBJ.
008010     ENTER LINKAGE.
008020     CALL 'WTSUB2' USING DIRA4 FORT-KEY.
008030     ENTER COBOL.
008040     ADD 1 TO NEXT-AV-SU2.
008050     GO TO A1.
008060 C.
```

```
008070      MOVE SUBJ-IN TO SUBJ-ISAM.
008080      MOVE NEXT-AV-SU1 TO NDEX-TO-SUB1.
008090      ENTER LINKAGE.
008100      CALL 'SUBSEQWT' USING DIRA5 SUB-TEST.
008110      ENTER COBOL.
008120      MOVE NEXT-AV-SU1 TO FORT-KEY.
008130      MOVE SUBJ-IN TO SUBJECT1.
008140      MOVE NEXT-AV-SU2 TO FRST-REC2.
008150      MOVE SPACES TO MESH.
008160      ENTER LINKAGE.
008170      CALL 'WTSUB1' USING DIRA3 FORT-KEY.
008180      ENTER COBOL.
008190      MOVE NEXT-AV-SU1 TO NEW-SUBJ.
008200      ADD 1 TO NEXT-AV-SU1.
008210      GO TO B3.
008220 D.
008230      MOVE OTHER-AUTH TO FORT-KEY.
008240      ENTER LINKAGE.
008250      CALL 'AUTFOR' USING DIRA6 FORT-KEY.
008251      ENTER COBOL.
009010      MOVE INDEX-T-ACCN TO ACN-IN FORT-KEY.
009020      GO TO A7.
009030 B2.
009040      MOVE NEXT-REC2 TO FORT-KEY.
009050      GO TO B1.
009060 A6.
009070      MOVE 1 TO FORT-KEY.
009080      ENTER LINKAGE.
009090      CALL 'WTACN' USING DIRA1 FORT-KEY.
009100      CLOSE SUBJECT-IN.
009110      ENTER LINKAGE.
009120      CALL 'CLSRN'.
009130      ENTER COBOL.
009140      STOP RUN.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'SUBRWRT'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 INPUT-OUTPUT SECTION.
001080 FILE-CONTROL.
001090     SELECT SUBJ-IN ASSIGN TO 'SYS011' DIRECT-ACCESS 2311
001100         ORGANIZATION IS INDEXED ACCESS IS RANDOM
001110         RECORD KEY IS SUBJ1
001120         SYMBOLIC KEY IS SUBJ-SYM-KEY.
001130 DATA DIVISION.
001140 FILE SECTION.
001150 FD  SUBJ-IN
001160     LABEL RECORDS ARE STANDARD
001170     RECORDING MODE IS F
001180     DATA RECORD IS SUBJ-REC.
001190 01  SUBJ-REC.
001200     02  FILLER                PICTURE X.
001210     02  SUBJ1                 PICTURE X(80).
001220     02  NDEX-FIRST-NAME       PICTURE 9(6).
001230 WORKING-STORAGE SECTION.
001240 77  SUBJ-SYM-KEY              PICTURE X(80).
001250 LINKAGE SECTION.
002010 01  S-KEY.
002020     02  SYM-KEY               PICTURE X(80).
002030 01  BUFER.
002040     02  ER                    PICTURE X.
002050     02  SUB                   PICTURE X(80).
002060     02  N-F-N                 PICTURE 9(6).
002070 PROCEDURE DIVISION.
002080     ENTER LINKAGE.
002090     ENTRY 'RWRSUB' USING BUFER S-KEY.
002100     ENTER COBOL.
002110     MOVE S-KEY TO SUBJ-SYM-KEY SUBJ1.
```

```
002120     MOVE N-F-N TO NDEX-FIRST-NAME.
002130     REWRITE SUBJ-REC INVALID KEY GO TO ERR.
002150     MOVE 0 TO ER.
002160 A1.
002170     ENTER LINKAGE.
002180     RETURN.
002190     ENTER COBOL.
002200 ERR.
002210     MOVE '1' TO ER.
002220     GO TO A1.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'SUBWRT'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 INPUT-OUTPUT SECTION.
001080 FILE-CONTROL.
001090     SELECT SUBJ-IN ASSIGN TO 'SYS012' DIRECT-ACCESS 2311
001100         ORGANIZATION IS INDEXED ACCESS IS RANDOM
001110         RECORD KEY IS SUBJ1
001120         SYMBOLIC KEY IS SUBJ-SYM-KEY.
001130 DATA DIVISION.
001140 FILE SECTION.
001150 FD  SUBJ-IN
001160     LABEL RECORDS ARE STANDARD
001170     RECORDING MODE IS F
001180     DATA RECORD IS SUBJ-REC.
001190 01  SUBJ-REC.
001200     02  FILLER                PICTURE X.
001210     02  SUBJ1                 PICTURE X(80).
001220     02  NDEX-FIRST-NAME       PICTURE 9(6).
001230 WORKING-STORAGE SECTION.
001240 77  SUBJ-SYM-KEY              PICTURE X(80).
001250 LINKAGE SECTION.
002010 01  S-KEY.
002020     02  SYM-KEY               PICTURE X(80).
002030 01  BUFER.
002040     02  ER                    PICTURE X.
002050     02  SUB                   PICTURE X(80).
002060     02  N-F-N                 PICTURE 9(6).
002070 PROCEDURE DIVISION.
002160 A1.
002170     ENTER LINKAGE.
002180     RETURN.
002190     ENTER COBOL.
```

```
002200 ERR.
002210     MOVE '1' TO ER.
002220     GO TO A1.
002230     ENTER LINKAGE.
002240     ENTRY 'SEQWTOPN'
002250     ENTER COBOL.
002260     OPEN OUTPUT SUBJ-IN.
002270     GO TO A1.
002280     ENTER LINKAGE.
002290     ENTRY 'SEQWTCLO'.
002300     ENTER COBOL.
002310     CLOSE SUBJ-IN.
002320     GO TO A1.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'SUBIO'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 INPUT-OUTPUT SECTION.
001080 FILE-CONTROL.
001090     SELECT SUBJ-IN ASSIGN TO 'SYS011' DIRECT-ACCESS 2311
001100         ORGANIZATION IS INDEXED ACCESS IS RANDOM
001110         RECORD KEY IS SUBJ1
001120         SYMBOLIC KEY IS SUBJ-SYM-KEY.
001130 DATA DIVISION.
001140 FILE SECTION.
001150 FD  SUBJ-IN
001160     LABEL RECORDS ARE STANDARD
001170     RECORDING MODE IS F
001180     DATA RECORD IS SUBJ-REC.
001190 01  SUBJ-REC.
001200     02  FILLER                PICTURE X.
001210     02  SUBJ1                 PICTURE X(80).
001220     02  NDEX-FIRST-NAME       PICTURE 9(6).
001230 WORKING-STORAGE SECTION.
001240 77  SUBJ-SYM-KEY              PICTURE X(80).
001250 LINKAGE SECTION.
002010 01  S-KEY.
002020     02  SYM-KEY               PICTURE X(80).
002030 01  BUFER.
002040     02  ER                    PICTURE X.
002050     02  SUB                   PICTURE X(80).
002060     02  N-F-N                 PICTURE 9(6).
002070 PROCEDURE DIVISION.
002080     ENTER LINKAGE.
002090     ENTRY 'SEQSUBIO' USING BUFER S-KEY.
002100     ENTER COBOL.
002110     MOVE S-KEY TO SUBJ-SYM-KEY SUB.
```

```
002120     READ SUBJ-IN INVALID KEY GO TO ERR.
002130     MOVE SUBJ1 TO SUB.
002140     MOVE NDEX-FIRST-NAME TO N-F-N.
002150     MOVE 0 TO ER.
002160 A1.
002170     ENTER LINKAGE.
002180     RETURN.
002190     ENTER COBOL.
002200 ERR.
002210     MOVE '1' TO ER.
002220     GO TO A1.
002230     ENTER LINKAGE.
002240     ENTRY 'SUBIOOPN'.
002250     ENTER COBOL.
003010     OPEN I-O SUBJ-IN.
003020     GO TO A1.
003030     ENTER LINKAGE.
003040     ENTRY 'SUBIOCLO'.
003050     ENTER COBOL.
003060     CLOSE SUBJ-IN.
003070     GO TO A1.
003080     ENTER LINKAGE.
003090     ENTRY 'RWRSUB' USING BUFER S-KEY.
003100     ENTER COBOL.
003110     MOVE S-KEY TO SUBJ-SYM-KEY SUBJ1.
003120     MOVE N-F-N TO NDEX-FIRST-NAME.
003130     REWRITE SUBJ-REC INVALID KEY GO TO ERR.
003140     MOVE 0 TO ER.
003150     GO TO A1.
003160     ENTER LINKAGE.
003170     ENTRY 'SUBSEQWT' USING BUFER S-KEY.
003180     ENTER COBOL.
003190     MOVE S-KEY TO SUBJ-SYM-KEY SUBJ1.
003200     MOVE N-F-N TO NDEX-FIRST-NAME.
003210     WRITE SUBJ-REC INVALID KEY GO TO ERR.
003220     MOVE 0 TO ER.
```

```
003230     GO TO A1.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'DROPAUT'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  STP1-PT-TO-ACCN          PICTURE 9(6) COMPUTATIONAL.
001100 77  STP2-PT-TO-AUT           PICTURE 9(6) COMPUTATIONAL.
001110 77  STP2-PT-TO-O-AUT         PICTURE 9(6) COMPUTATIONAL.
001120 77  STP2-PT-TO-NEXT-ACN      PICTURE 9(6) COMPUTATIONAL.
001130 77  STP3-PT-TO-ACN           PICTURE 9(6) COMPUTATIONAL.
001140 77  STP3-THIS-REC-NO         PICTURE 9(6) COMPUTATIONAL.
001150 77  STP3-NEXT-AUT-REC        PICTURE 9(6) COMPUTATIONAL.
001160 77  STP6-THIS-REC-NO         PICTURE 9(6) COMPUTATIONAL.
001170 77  STP6-PT-TO-AUT           PICTURE 9(6) COMPUTATIONAL.
001180 77  STP6-PT-TO-O-AUT         PICTURE 9(6) COMPUTATIONAL.
001190 77  STP6-PT-TO-NEXT-ACN      PICTURE 9(6) COMPUTATIONAL.
001200 77  STP21-PT-TO-FIRST-AUT    PICTURE 9(6) COMPUTATIONAL.
001210 77  FORT-KEY                 PICTURE 9(6) COMPUTATIONAL.
001220 77  RLDARN                   PICTURE 9(6) COMPUTATIONAL.
001230 01  SYM-KEY.
001240         02  KEY-ARAY         PICTURE X OCCURS 213 TIMES.
001250 01  SUB                      PICTURE 9(3) VALUE ZERO.
001260 01  HIVAL                    PICTURE X VALUE IS HIGH-VALUE.
002010 LINKAGE SECTION.
002020 77  RTLIST                   PICTURE 9(6) COMPUTATIONAL.
002030 01  AUTH.
002040         02  AUTH-ARAY        PICTURE X OCCURS 213 TIMES.
002050 01  ACN-NO                   PICTURE X(16).
002060 01  DIRA1.
002070         02  ER1              PICTURE X.
002080         02  DESC-SEEK-KEY    PICTURE X(16).
002090         02  MIDL             PICTURE X(412).
002100         02  NDEX-TO-ACCN     PICTURE 9(6).
```

```
002110     02  NINS                 PICTURE 9(12).
002120     02  LSTS                 PICTURE X(43).
002130 01  DIRA2 REDEFINES DIRA1.
002140     02  ER2                  PICTURE X.
002150     02  AUTH2                PICTURE X(213).
002160     02  NDEX-FIRST-NAME      PICTURE 9(6).
002170     02  FILLER               PICTURE X(270).
002180 01  DIRA3 REDEFINES DIRA2.
002190     02  AUTH3                PICTURE X(213).
002200     02  NDEX-NEXT-AUT        PICTURE 9(6) COMPUTATIONAL.
002210     02  NDEX-ACCN-A          PICTURE 9(6) COMPUTATIONAL.
002220     02  FILLER               PICTURE X(269).
002230 01  DIRA4 REDEFINES DIRA3.
002240     02  ACN-NO4              PICTURE X(16).
002250     02  N-T-T1               PICTURE 9(6) COMPUTATIONAL.
003010     02  N-T-S1               PICTURE 9(6) COMPUTATIONAL.
003020     02  NDEX-TO-AUT          PICTURE 9(6) COMPUTATIONAL.
003030     02  NDEX-OTHER-AUT       PICTURE 9(6) COMPUTATIONAL.
003040     02  FILLER               PICTURE X(4).
003050     02  NDEX-NEXT-ACN        PICTURE 9(6) COMPUTATIONAL.
003060     02  FILLER               PICTURE X(450).
003070 PROCEDURE DIVISION.
003080     ENTER LINKAGE.
003090     ENTRY 'DROPA' USING AUTH ACN-NO DIRA1 RTLIST.
003100     ENTER COBOL.
003110     MOVE 3 TO RLDARN.
003120     ENTER LINKAGE.
003130     CALL 'IOENQ' USING RLDARN RTLIST DIRA1 ACN-NO.
003140     ENTER COBOL.
003141     IF ER1 = '1' GO TO RTRN2.
003150     MOVE NDEX-TO-ACCN TO STP1-PT-TO-ACCN FORT-KEY.
003160     MOVE 4 TO RLDARN.
003170     ENTER LINKAGE.
003180     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
003190     ENTER COBOL.
003200     MOVE NDEX-TO-AUT TO STP2-PT-TO-AUT FORT-KEY.
```

```
003210      MOVE NDEX-OTHER-AUT TO STP2-PT-TO-O-AUT.
003220      MOVE NDEX-NEXT-ACN TO STP2-PT-TO-NEXT-ACN.
003221      MOVE FORT-KEY TO STP3-THIS-REC-NO.
003230      MOVE 5 TO RLDARN.
003240      ENTER LINKAGE.
003250      CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
004010      ENTER COBOL.
004020      MOVE NDEX-ACCN-A TO STP3-PT-TO-ACN.
004050      MOVE NDEX-NEXT-AUT TO STP3-NEXT-AUT-REC.
004060  STEP4.
004070      IF AUTH3 = AUTH GO TO STEP6.
004080      IF STP2-PT-TO-AUT = STP2-PT-TO-O-AUT GO TO RTRN1.
004090      MOVE 5 TO RLDARN.
004100      MOVE STP2-PT-TO-O-AUT TO FORT-KEY.
004101      MOVE FORT-KEY TO STP3-THIS-REC-NO.
004110      ENTER LINKAGE.
004120      CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
004121      ENTER COBOL.
004130      MOVE AUTH3 TO SYM-KEY.
004140      MOVE NDEX-ACCN-A TO STP3-PT-TO-ACN.
004170      MOVE NDEX-NEXT-AUT TO STP3-NEXT-AUT-REC.
004180      MOVE 4 TO RLDARN.
004190      MOVE STP3-PT-TO-ACN TO FORT-KEY.
004200      ENTER LINKAGE.
004210      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
004220      ENTER COBOL.
004230      MOVE NDEX-OTHER-AUT TO STP2-PT-TO-O-AUT.
004240      MOVE NDEX-NEXT-ACN TO STP2-PT-TO-NEXT-ACN.
004250      MOVE SYM-KEY TO AUTH3. GO TO STEP4.
005010  STEP6.
005020      MOVE 4 TO RLDARN.
005030      MOVE STP3-PT-TO-ACN TO FORT-KEY.
005040  SA. MOVE FORT-KEY TO STP6-THIS-REC-NO.
005041      ENTER LINKAGE.
005050      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
005060      ENTER COBOL.
```

```
005090      MOVE NDEX-TO-AUT TO STP6-PT-TO-AUT.
005100      MOVE NDEX-OTHER-AUT TO STP6-PT-TO-O-AUT.
005110      MOVE NDEX-NEXT-ACN TO STP6-PT-TO-NEXT-ACN.
005120      IF ACN-NO4 = ACN-NO GO TO STEP9.
005130      IF STP6-PT-TO-NEXT-ACN = 0 GO TO RTRN2.
005140      MOVE STP6-PT-TO-NEXT-ACN TO FORT-KEY.
005150      GO TO SA.
005160 STEP9.
005170      IF STP6-PT-TO-NEXT-ACN = 0 GO TO STEP15.
005180 ST.  MOVE 32 TO RLDARN.
005190      MOVE STP6-THIS-REC-NO TO FORT-KEY.
005200      MOVE HIVAL TO ACN-NO4.
005210      ENTER LINKAGE.
005220      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
005230      ENTER COBOL.
005240      MOVE 5 TO RLDARN.
005250      MOVE STP3-THIS-REC-NO TO FORT-KEY.
006010      ENTER LINKAGE.
006020      CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
006030      ENTER COBOL.
006040      MOVE STP6-PT-TO-NEXT-ACN TO NDEX-ACCN-A.
006041      MOVE STP3-THIS-REC-NO TO FORT-KEY.
006050      MOVE 27 TO RLDARN.
006070      ENTER LINKAGE.
006080      CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
006090      ENTER COBOL.
006100      MOVE 4 TO RLDARN.
006110      MOVE STP1-PT-TO-ACCN TO FORT-KEY.
006120      ENTER LINKAGE.
006130      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
006140      ENTER COBOL.
006150      MOVE STP6-PT-TO-O-AUT TO NDEX-OTHER-AUT.
006160      MOVE 32 TO RLDARN.
006161      MOVE STP1-PT-TO-ACCN TO FORT-KEY.
006180      ENTER LINKAGE.
006190      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
```

```
006200      ENTER COBOL.
007010      ENTER LINKAGE.
007020      RETURN.
007030      ENTER COBOL.
007040 STEP15.
007050      IF STP3-PT-TO-ACN IS NOT EQUAL TO STP6-THIS-REC-NO GO TO ST.
007060      MOVE 32 TO RLDARN.
007070      MOVE STP6-THIS-REC-NO TO FORT-KEY.
007080      MOVE HIVAL TO ACN-NO4.
007090      ENTER LINKAGE.
007100      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
007110      ENTER COBOL.
007120      MOVE 27 TO RLDARN.
007130      MOVE STP3-THIS-REC-NO TO FORT-KEY.
007140      MOVE HIVAL TO AUTH3.
007150      ENTER LINKAGE.
007160      CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
007170      ENTER COBOL.
007180      MOVE '1' TO ER2.
007190      MOVE 2_ TO RLDARN.
007191      MOVE SPACES TO SYM-KEY.
007192      MOVE 1 TO SUB.
007193 LOOP-GO.
007194      IF AUTH-ARAY (SUB) = '.' GO TO LOOP-STOP.
007195      MOVE AUTH-ARAY (SUB) TO KEY-ARAY (SUB).
007196      ADD 1 TO SUB.
007197      IF SUB   213 GO TO LOOP-STOP.
007198      GO TO LOOP-GO.
007199 LOOP-STOP.
007200      ENTER LINKAGE.
008010      CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
008020      ENTER COBOL.
008030      IF NDEX-FIRST-NAME = STP3-THIS-REC-NO GO TO STEP20.
008040      MOVE NDEX-FIRST-NAME TO STP21-PT-TO-FIRST-AUT.
008050      MOVE 2 TO ER2.
008060      ENTER LINKAGE.
```

```
008070     CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
008080     ENTER COBOL.
008090     MOVE STP21-PT-TO-FIRST-AUT TO FORT-KEY.
008100 S7. MOVE 5 TO RLDARN.
008110     ENTER LINKAGE.
008120     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
008130     ENTER COBOL.
008140     IF NDEX-NEXT-AUT = STP3-THIS-REC-NO GO TO STEP24.
008150     MOVE NDEX-NEXT-AUT TO FORT-KEY.
008160     GO TO S7.
008170 STEP24.
008180     MOVE STP3-NEXT-AUT-REC TO NDEX-NEXT-AUT.
008190     MOVE 27 TO RLDARN.
009010     ENTER LINKAGE.
009020     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
009030     ENTER COBOL.
009040     ENTER LINKAGE.
009050     RETURN.
009060     ENTER COBOL.
009070 STEP20.
009080     MOVE HIVAL TO ER2.
009090     ENTER LINKAGE.
009100     CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
009110     ENTER COBOL.
009120     ENTER LINKAGE.
009130     RETURN.
009140     ENTER COBOL.
009150 RTRN1.
009160     MOVE '8' TO ER1
009170 R.  ENTER LINKAGE.
009180     RETURN.
009190     ENTER COBOL.
009200 RTRN2.
009210     MOVE '9' TO ER1.
009220     GO TO R.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'ADDAUT'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  SWIT                         PICTURE X.
001091 77  RLDARN                       PICTURE 9(6) COMPUTATIONAL.
001100 77  NEXT-AVAIL-ACCN              PICTURE 9(6) COMPUTATIONAL.
001110 77  FORT-KEY                     PICTURE 9(6) COMPUTATIONAL.
001120 77  ACCN-HOLD-3                  PICTURE 9(6) COMPUTATIONAL.
001130 77  AUTH-HOLD-5                  PICTURE 9(6) COMPUTATIONAL.
001140 77  OTHER-AUTH-HOLD-5            PICTURE 9(6) COMPUTATIONAL.
001150 77  KEY-HOLD-14                  PICTURE 9(6) COMPUTATIONAL.
001160 77  NEXT-AVAIL-AUTH-REC          PICTURE 9(6) COMPUTATIONAL.
001170 77  LAST-AUT-REC                 PICTURE 9(6) COMPUTATIONAL.
001180 77  LAST-ACCN-REC                PICTURE 9(6) COMPUTATIONAL.
001181 77  FORT-KEY-HOLD                PICTURE 9(6) COMPUTATIONAL.
001190 01  SYM-KEY.
001200         02  KEY-ARAY             PICTURE X OCCURS 213 TIMES.
002010 01  SUB                          PICTURE 9(3)  VALUE ZERO.
002020 LINKAGE SECTION.
002030 77  RTLIST                       PICTURE 9(6) COMPUTATIONAL.
002040 01  AUTH-ARG.
002050         02  AUTH-ARAY            PICTURE X OCCURS 213 TIMES.
002060 01  ACN-ARG                      PICTURE X(16).
002070 01  DIRA1.
002080         02  ER1                  PICTURE X.
002090         02  DESC-SEEK-KEY        PICTURE X(16).
002100         02  MIDL                 PICTURE X(412).
002110         02  NDEX-TO-ACCN         PICTURE 9(6).
002120         02  NINS                 PICTURE 9(12).
002130         02  LSTS                 PICTURE X(43).
002140 01  DIRA2 REDEFINES DIRA1.
```

```
002150     02  ER2                    PICTURE X.
002160     02  AUTH-ISAM              PICTURE X(213).
002170     02  NDEX-TO-FRST-NAME      PICTURE 9(6).
002180     02  FILLER                 PICTURE X(270).
002190 01  DIRA3 REDEFINES DIRA2.
002200     02  AUTH-BUF               PICTURE X(213).
003010     02  NDEX-TO-NEXT-AUT-REC PICTURE 9(6) COMPUTATIONAL.
003020     02  NDEX-TO-ACCN-A         PICTURE 9(6) COMPUTATIONAL.
003021     02  SEE-AUTH               PICTURE 9(6) COMPUTATIONAL.
003030     02  FILLER                 PICTURE X(265).
003040 01  DIRA4 REDEFINES DIRA3.
003050     02  ACN-NO                 PICTURE X(16).
003060     02  NDEX-TO-TITL1          PICTURE 9(6) COMPUTATIONAL.
003070     02  NDEX-TO-SUBJ           PICTURE 9(6) COMPUTATIONAL.
003080     02  NDEX-TO-AUTH           PICTURE 9(6) COMPUTATIONAL.
003090     02  NDEX-TO-O-AUTH         PICTURE 9(6) COMPUTATIONAL.
003100     02  AUTH-LEVEL             PICTURE 9(6) COMPUTATIONAL.
003110     02  NDEX-TO-NEXT-ACCN      PICTURE 9(6) COMPUTATIONAL.
003120     02  FILLER                 PICTURE X(450).
003130 PROCEDURE DIVISION.
003140     ENTER LINKAGE.
003150     ENTRY 'ADDA' USING AUTH-ARG ACN-ARG DIRA1 RTLIST.
003160     ENTER COBOL.
003170     MOVE ACN-ARG TO SYM-KEY.
003180     MOVE '0' TO SWIT.
003190     MOVE 3 TO RLDARN.
003200     ENTER LINKAGE.
004010     CALL 'IOENQ' USING RLDARN RTLIST DIRA1 SYM-KEY.
004020     ENTER COBOL.
004030     IF NDEX-TO-ACCN IS NOT EQUAL TO 0 GO TO THRE.
004040     MOVE 5 TO FORT-KEY.
004050     MOVE 26 TO RLDARN.
004060     ENTER LINKAGE.
004070     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
004080     ENTER COBOL.
004090     MOVE FORT-KEY TO NEXT-AVAIL-ACCN.
```

```
004100      MOVE 25 TO RLDARN.
004110      MOVE '1' TO ER1.
004120      MOVE ACN-ARG TO SYM-KEY.
004130      ENTER LINKAGE.
004140      CALL 'IOENQ' USING RLDARN RTLIST DIRA1 SYM-KEY.
004150      ENTER COBOL.
004160      MOVE NEXT-AVAIL-ACCN TO NDEX-TO-ACCN.
004170      MOVE '2' TO ER1.
004180      ENTER LINKAGE.
004190      CALL 'IOENQ' USING RLDARN RTLIST DIRA1 SYM-KEY.
004200      ENTER COBOL.
005010      MOVE NEXT-AVAIL-ACCN TO FORT-KEY FORT-KEY-HOLD.
005020      MOVE 33 TO RLDARN.
005030      ENTER LINKAGE.
005040      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
005050      ENTER COBOL.
005060      MOVE FORT-KEY-HOLD TO FORT-KEY.
005070      MOVE ACN-ARG TO ACN-NO.
005080      MOVE 0 TO NDEX-TO-TITL1 NDEX-TO-SUBJ NDEX-TO-AUTH
005090      NDEX-TO-O-AUTH NDEX-TO-NEXT-ACCN.
005100      ENTER LINKAGE.
005110      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
005120      ENTER COBOL.
005130      MOVE NEXT-AVAIL-ACCN TO NDEX-TO-ACCN.
005140 THRE.
005150      MOVE NDEX-TO-ACCN TO FORT-KEY ACCN-HOLD-3 FORT-KEY-HOLD.
005160      MOVE 4 TO RLDARN.
005170      ENTER LINKAGE.
005180      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
005190      ENTER COBOL.
005200      MOVE NDEX-TO-AUTH TO AUTH-HOLD-5.
006010      MOVE NDEX-TO-O-AUTH TO OTHER-AUTH-HOLD-5.
006020      IF NDEX-TO-AUTH = 0 GO TO SIX-ONE.
006030      IF NDEX-TO-AUTH = NDEX-TO-O-AUTH GO TO FORTEEN.
006040 AT.  MOVE NDEX-TO-O-AUTH TO FORT-KEY.
006050      MOVE 5 TO RLDARN.
```

```
006060     ENTER LINKAGE.
006070     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
006080     ENTER COBOL.
006090     MOVE NDEX-TO-ACCN-A TO FORT-KEY FORT-KEY-HOLD.
006100 AU. MOVE 4 TO RLDARN.
006110     ENTER LINKAGE.
006120     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
006130     ENTER COBOL.
006140     IF ACN-NO IS NOT EQUAL TO ACN-ARG GO TO TWLV-ONE.
006150 TWLV.
006160     IF NDEX-TO-D-AUTH = AUTH-HOLD-5 GO TO FORTEEN.
006170     GO TO AT.
006180 TWLV-ONE.
006190     MOVE NDEX-TO-NEXT-ACCN TO FORT-KEY FORT-KEY-HOLD.
006200     GO TO AU.
007010 FORTEEN.
007020     MOVE FORT-KEY-HOLD TO FORT-KEY.
007030     MOVE FORT-KEY TO KEY-HOLD-14. MOVE SPACES TO SYM-KEY.
007040 AV. ADD 1 TO SUB.
007050     IF SUB   213 GO TO SIX-TEEN.
007060     IF AUTH-ARAY (SUB) = '.' GO TO SIX-TEEN.
007070     MOVE AUTH-ARAY (SUB) TO KEY-ARAY (SUB).
007080     GO TO AV.
007090 SIX-TEEN.
007100     MOVE 0 TO SUB.
007110     MOVE 0 TO RLDARN.
007120     ENTER LINKAGE.
007130     CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
007140     ENTER COBOL.
007150     IF ER1 = '1' GO TO A.
007160     MOVE NDEX-TO-FRST-NAME TO FORT-KEY.
007170 NIN-TEEN.
007180     MOVE 5 TO RLDARN.
007181     MOVE FORT-KEY TO FORT-KEY-HOLD.
007190     ENTER LINKAGE.
007200     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
```

```
008010      ENTER COBOL.
008020      IF AUTH-ARG = AUTH-BUF GO TO TWO-ONE-ONE.
008030      IF NDEX-TO-NEXT-AUT-REC = 0 GO TO TWO-THREE.
008040      MOVE NDEX-TO-NEXT-AUT-REC TO FORT-KEY.
008050      GO TO NIN-TEEN.
008060 TWO-ONE-ONE.
008070      MOVE FORT-KEY-HOLD TO FORT-KEY.
008080      MOVE FORT-KEY TO NEXT-AVAIL-AUTH-REC.
008090      MOVE NDEX-TO-ACCN-A TO FORT-KEY.
008100      GO TO C.
008110 TWO-THREE.
008120      MOVE FORT-KEY-HOLD TO FORT-KEY.
008130      MOVE FORT-KEY TO LAST-AUT-REC.
008140      GO TO B.
008150 SIX-ONE.
008160      MOVE '1' TO SWIT.
008170      GO TO FORTEEN.
008180 A. MOVE 1 TO FORT-KEY.
008190      MOVE 26 TO RLDARN.
008200      ENTER LINKAGE.
009010      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
009020      ENTER COBOL.
009030      MOVE FORT-KEY TO NEXT-AVAIL-AUTH-REC.
009031      MOVE '3' TO ER2.
009040      MOVE NEXT-AVAIL-AUTH-REC TO NDEX-TO-FRST-NAME.
009050      MOVE SYM-KEY TO AUTH-ISAM.
009060      MOVE 22 TO RLDARN.
009070      ENTER LINKAGE.
009080      CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
009090      ENTER COBOL.
009100 A6. MOVE KEY-HOLD-14 TO FORT-KEY FORT-KEY-HOLD.
009110      MOVE 33 TO RLDARN.
009120      ENTER LINKAGE.
009130      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
009140      ENTER COBOL.
009150      MOVE FORT-KEY-HOLD TO FORT-KEY.
```

```
010151      MOVE 0 TO SEE-AUTH.
009160      MOVE NEXT-AVAIL-AUTH-REC TO NDEX-TO-O-AUTH.
009170      ENTER LINKAGE.
009180      CALL 'IDENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
009190      ENTER COBOL.
009200      IF SWIT IS NOT EQUAL TO '1' GO TO A10.
010010      MOVE ACCN-HOLD-3 TO NEXT-AVAIL-ACCN.
010020      MOVE NEXT-AVAIL-AUTH-REC TO AUTH-HOLD-5.
010030      GO TO A12.
010040 A10.
010050      MOVE 5 TO FORT-KEY.
010060      MOVE 26 TO RLDARN.
010070      ENTER LINKAGE.
010080      CALL 'IDENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
010090      ENTER COBOL.
010100      MOVE FORT-KEY TO NEXT-AVAIL-ACCN.
010110 A12.
010120      MOVE NEXT-AVAIL-AUTH-REC TO FORT-KEY.
010130      MOVE AUTH-ARG TO AUTH-BUF.
010140      MOVE 0 TO NDEX-TO-NEXT-AUT-REC.
010150      MOVE NEXT-AVAIL-ACCN TO NDEX-TO-ACCN-A.
010160      MOVE 27 TO RLDARN.
010170      ENTER LINKAGE.
010180      CALL 'IDENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
010190      ENTER COBOL.
010200 A14.
011010      MOVE NEXT-AVAIL-ACCN TO FORT-KEY.
011020      MOVE ACN-ARG TO ACN-NO.
011030      MOVE 0 TO NDEX-TO-TITL1 NDEX-TO-SUBJ.
011040      MOVE NEXT-AVAIL-AUTH-REC TO NDEX-TO-AUTH.
011050      MOVE AUTH-HOLD-5 TO NDEX-TO-O-AUTH.
011060      MOVE 0 TO NDEX-TO-NEXT-ACCN.
011061      MOVE 1 TO AUTH-LEVEL.
011070      MOVE 32 TO RLDARN.
011080      ENTER LINKAGE.
011090      CALL 'IDENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
```

```
011100     ENTER COBOL.
011110 R.  ENTER LINKAGE.
011120     RETURN.
011130     ENTER COBOL.
011140 B.  MOVE 1 TO FORT-KEY.
011150     MOVE 26 TO RLDARN.
011160     ENTER LINKAGE.
011170     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
011180     ENTER COBOL.
011190     MOVE FORT-KEY TO NEXT-AVAIL-AUTH-REC.
011200     MOVE LAST-AUT-REC TO FORT-KEY FORT-KEY-HOLD.
012010     MOVE 34 TO RLDARN.
012020     ENTER LINKAGE.
012030     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
012040     ENTER COBOL.
012050     MOVE NEXT-AVAIL-AUTH-REC TO NDEX-TO-NEXT-AUT-REC.
012060     MOVE FORT-KEY-HOLD TO FORT-KEY.
012070     ENTER LINKAGE.
012080     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
012090     ENTER COBOL.
012100     GO TO A6.
012110 C.  MOVE 4 TO RLDARN.
012111     MOVE FORT-KEY TO FORT-KEY-HOLD.
012120     ENTER LINKAGE.
012130     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
012140     ENTER COBOL.
012150     IF ACN-ARG = ACN-NO GO TO R.
012160     IF NDEX-TO-NEXT-ACCN = 0 GO TO C4.
012170     MOVE NDEX-TO-NEXT-ACCN TO FORT-KEY.
012180     GO TO C.
012190 C4. MOVE FORT-KEY-HOLD TO FORT-KEY.
012200     MOVE FORT-KEY TO LAST-ACCN-REC.
013010     IF SWIT = '1' GO TO C7.
013020     MOVE 5 TO FORT-KEY.
013030     MOVE 26 TO RLDARN.
013040     ENTER LINKAGE.
```

```
013050     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
013060     ENTER COBOL.
013070     MOVE FORT-KEY TO NEXT-AVAIL-ACCN.
013080     MOVE LAST-ACCN-REC TO FORT-KEY FORT-KEY-HOLD.
013090 C9. MOVE 33 TO RLDARN.
013100     ENTER LINKAGE.
013110     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
013120     ENTER COBOL.
013130     MOVE FORT-KEY-HOLD TO FORT-KEY.
013140     MOVE NEXT-AVAIL-ACCN TO NDEX-TO-NEXT-ACCN.
013150     ENTER LINKAGE.
013160     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
013170     ENTER COBOL.
013180     MOVE KEY-HOLD-14 TO FORT-KEY FORT-KEY-HOLD.
013190     MOVE 33 TO RLDARN.
013200     ENTER LINKAGE.
014010     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
014020     ENTER COBOL.
014030     MOVE FORT-KEY-HOLD TO FORT-KEY.
014040     MOVE NEXT-AVAIL-AUTH-REC TO NDEX-TO-O-AUTH.
014050     ENTER LINKAGE.
014060     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
014070     ENTER COBOL.
014080     GO TO A14.
014090 C7. MOVE LAST-ACCN-REC TO FORT-KEY FORT-KEY-HOLD.
014100     MOVE ACCN-HOLD-3 TO NEXT-AVAIL-ACCN.
014110     MOVE NEXT-AVAIL-AUTH-REC TO AUTH-HOLD-5.
014120     GO TO C9.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'DROPSUBJ'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  FORT-KEY                 PICTURE 9(6) COMPUTATIONAL.
001100 77  ACCN-HOLD-3              PICTURE 9(6) COMPUTATIONAL.
001110 77  TITL-HOLD-5              PICTURE 9(6) COMPUTATIONAL.
001120 77  TITL1-ONE-BACK           PICTURE 9(6) COMPUTATIONAL.
001130 77  TITL1-HOLD-7             PICTURE 9(6) COMPUTATIONAL.
001140 77  FRST-REC-HOLD-9          PICTURE 9(6) COMPUTATIONAL.
001150 77  FRST-REC2-HOLD-18        PICTURE 9(6) COMPUTATIONAL.
001160 77  BACK-ONE-22              PICTURE 9(6) COMPUTATIONAL.
001170 77  HOLD-THIS-REC-24         PICTURE 9(6) COMPUTATIONAL.
001180 77  HOLD-NEXT2-25            PICTURE 9(6) COMPUTATIONAL.
001190 77  HOLD-OTHER-25            PICTURE 9(6) COMPUTATIONAL.
001200 77  HOLD-REC2-36             PICTURE 9(6) COMPUTATIONAL.
002010 77  HOLD-48                  PICTURE 9(6) COMPUTATIONAL.
002011 77  RLDARN                   PICTURE 9(6) COMPUTATIONAL.
002020 01  SYM-KEY                  PICTURE X(240).
002030 01  HIVAL                    PICTURE X VALUE IS HIGH-VALUE.
002040 LINKAGE SECTION.
002050 77  RTLIST                   PICTURE 9(6) COMPUTATIONAL.
002060 01  TITLE-ARG                PICTURE X(80).
002070 01  ACN-NO                   PICTURE X(16).
002080 01  DIRA1.
002090         02  ER1              PICTURE X.
002100         02  DESC-SEEK-KEY    PICTURE X(16).
002110         02  MIDL             PICTURE X(412).
002120         02  NDEX-TO-ACCN     PICTURE 9(6).
002130         02  NINS             PICTURE 9(12).
002140         02  LSTS             PICTURE X(43).
002150 01  DIRA2 REDEFINES DIRA1.
```

```
002160     02  ER2                     PICTURE X.
002170     02  TITLE-ISAM              PICTURE X(80).
002180     02  NDEX-TO-TITL1           PICTURE 9(6).
002190     02  FILLER                  PICTURE X(243).
002200 01  DIRA3 REDEFINES DIRA2.
003010     02  TITLE-BUF               PICTURE X(80).
003020     02  NDEX-FRST-REC2          PICTURE 9(6) COMPUTATIONAL.
003030     02  FILLER                  PICTURE X(246).
003040 01  DIRA4 REDEFINES DIRA3.
003050     02  NDEX-TO-ACCN-T          PICTURE 9(6) COMPUTATIONAL.
003060     02  NDEX-TO-NEXT-REC2       PICTURE 9(6) COMPUTATIONAL.
003070     02  NDEX-TO-OTHER-TITL      PICTURE 9(6) COMPUTATIONAL.
003080     02  FILLER                  PICTURE X(478).
003090 01  DIRA5 REDEFINES DIRA4.
003100     02  ACK-NO                  PICTURE X(16).
003110     02  NDEX-TO-TITL            PICTURE 9(6) COMPUTATIONAL.
003111     02  NDEX-TO-SUBJ            PICTURE 9(6) COMPUTATIONAL.
003120     02  FILLER                  PICTURE X(466).
003130 PROCEDURE DIVISION.
003140     ENTER LINKAGE.
003150     ENTRY 'DROPS' USING TITLE-ARG ACN-NO DIRA1 RTLIST.
003160     ENTER COBOL.
003170     MOVE ACN-NO TO SYM-KEY DESC-SEEK-KEY.
003180     MOVE 3 TO RLDARN.
003190     ENTER LINKAGE.
003200     CALL 'IOENQ' USING RLDARN RTLIST DIRA1 SYM-KEY.
004010     ENTER COBOL.
004020     MOVE NDEX-TO-ACCN TO FORT-KEY ACCN-HOLD-3.
004030     MOVE 4 TO RLDARN.
004040     ENTER LINKAGE.
004050     CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
004060     ENTER COBOL.
004070     MOVE NDEX-TO-SUBJ TO FORT-KEY TITL-HOLD-5.
004080     MOVE 8 TO RLDARN.
004090     ENTER LINKAGE.
004100     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
```

```
004110      ENTER COBOL.
004120      MOVE TITL-HOLD-5 TO TITL1-ONE-BACK TITL1-HOLD-7.
004130 ATE.
004140      IF TITLE-BUF = TITLE-ARG GO TO A.
004150      MOVE NDEX-FRST-REC2 TO FORT-KEY FRST-REC-HOLD-9.
004160 TIN.
004170      MOVE 9 TO RLDARN.
004180      ENTER LINKAGE.
004190      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
004200      ENTER COBOL.
005010      IF NDEX-TO-ACCN-T = ACCN-HOLD-3 GO TO FITEEN.
005020      IF NDEX-TO-NEXT-REC2 = 0 GO TO RTRN8.
005030      MOVE NDEX-TO-NEXT-REC2 TO FORT-KEY.
005040      GO TO TIN.
005050 FITEEN.
005060      IF NDEX-TO-OTHER-TITL = TITL-HOLD-5 GO TO RTRN8.
005070      MOVE NDEX-TO-OTHER-TITL TO FORT-KEY.
005080      MOVE TITL1-HOLD-7 TO TITL1-ONE-BACK.
005090      MOVE NDEX-TO-OTHER-TITL TO TITL1-HOLD-7.
005100      MOVE 8 TO RLDARN.
005110      ENTER LINKAGE.
005120      CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
005130      ENTER COBOL.
005140      GO TO ATE.
005150 A.
005160      MOVE NDEX-FRST-REC2 TO FORT-KEY FRST-REC2-HOLD-18.
005170      MOVE 0 TO BACK-ONE-22.
005180 NINTEEN.
005190      MOVE 9 TO RLDARN.
005200      ENTER LINKAGE.
006010      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
006020      ENTER COBOL.
006030      IF NDEX-TO-ACCN-T = ACCN-HOLD-3 GO TO TWNTY3.
006040      IF NDEX-TO-NEXT-REC2 = 0 GO TO RTRN8.
006060      MOVE FORT-KEY TO BACK-ONE-22.
006070      MOVE NDEX-TO-NEXT-REC2 TO FORT-KEY.
```

```
006080     GO TO NINTEEN.
006090 TWNTY3.
006100     IF BACK-ONE-22 = 0 GO TO B.
006120     MOVE FORT-KEY TO HOLD-THIS-REC-24.
006130     MOVE NDEX-TO-NEXT-REC2 TO HOLD-NEXT2-25.
006140     MOVE NDEX-TO-OTHER-TITL TO HOLD-OTHER-25.
006150     MOVE BACK-ONE-22 TO FORT-KEY. MOVE 38 TO RLDARN.
006160     ENTER LINKAGE.
006170     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
006180     ENTER COBOL.
006200     MOVE HOLD-NEXT2-25 TO NDEX-TO-NEXT-REC2.
007010     ENTER LINKAGE.
007020     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
007030     ENTER COBOL.
007040     MOVE FRST-REC-HOLD-9 TO FORT-KEY.
007050     MOVE 38 TO RLDARN.
007060     ENTER LINKAGE.
007070     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
007080     ENTER COBOL.
007090     MOVE HOLD-OTHER-25 TO NDEX-TO-OTHER-TITL.
007110     ENTER LINKAGE.
007120     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
007130     ENTER COBOL.
007140 C.
007150     IF TITL1-ONE-BACK = TITL1-HOLD-7 GO TO FOUR-NINE.
007160     MOVE HIVAL TO ACK-NO.
007170     MOVE HOLD-THIS-REC-24 TO FORT-KEY.
007180     MOVE 29 TO RLDARN.
007190     ENTER LINKAGE.
007200     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
008010     ENTER COBOL.
008020 R.  MOVE '0' TO ACK-NO.
008030     ENTER LINKAGE.
008040     RETURN.
008050     ENTER COBOL.
008060 FOUR-NINE.
```

```
008070     MOVE HOLD-THIS-REC-24 TO FORT-KEY.
008080     MOVE 38 TO RLDARN.
008090     ENTER LINKAGE.
008100     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
008110     ENTER COBOL.
008130     MOVE HIVAL TO ACK-NO.
008140     MOVE NDEX-TO-OTHER-TITL TO HOLD-48.
008150     ENTER LINKAGE.
008160     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
008170     ENTER COBOL.
008180     MOVE ACCN-HOLD-3 TO FORT-KEY.
008190     MOVE 33 TO RLDARN.
008200     ENTER LINKAGE.
009010     CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
009020     ENTER COBOL.
009040     IF HOLD-48 = NDEX-TO-SUBJ GO TO FIVE-7.
009050     MOVE HOLD-48 TO NDEX-TO-SUBJ.
009060 FIVE-9.
009070     ENTER LINKAGE.
009080     CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
009090     ENTER COBOL.
009100     GO TO R.
009110 FIVE-7.
009120     MOVE 0 TO NDEX-TO-SUBJ.
009130     GO TO FIVE-9.
009140 B.
009150     IF NDEX-TO-NEXT-REC2 = 0 GO TO THREE-7.
009160     MOVE TITL1-HOLD-7 TO FORT-KEY.
009170     MOVE NDEX-TO-NEXT-REC2 TO HOLD-REC2-36.
009180     MOVE 37 TO RLDARN.
009190     ENTER LINKAGE.
009200     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
010010     ENTER COBOL.
010020     MOVE HOLD-REC2-36 TO NDEX-FRST-REC2.
010040     ENTER LINKAGE.
010050     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
```

```
010060     ENTER COBOL.
010070 D.
010080     MOVE FRST-REC2-HOLD-18 TO HOLD-THIS-REC-24.
010090     GO TO C.
010100 THREE-7.
010110     MOVE TITLE-ARG TO SYM-KEY.
010120     MOVE '1' TO ER1.
010130     MOVE 24 TO RLDARN.
010140     ENTER LINKAGE.
010150     CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
010160     ENTER COBOL.
010170     MOVE HIVAL TO ER1.
010180     MOVE TITLE-ARG TO SYM-KEY.
010190     ENTER LINKAGE.
010200     CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
011010     ENTER COBOL.
011020     MOVE TITL1-HOLD-7 TO FORT-KEY.
011030     MOVE HIVAL TO TITLE-BUF.
011040     MOVE 28 TO RLDARN.
011050     ENTER LINKAGE.
011060     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
011070     ENTER COBOL.
011080     GO TO D.
011090 RTRN8.
011100     MOVE '8' TO ACK-NO.
011110     ENTER LINKAGE.
011120     RETURN.
011130     ENTER COBOL.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'ADDSUB'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  SWIT                        PICTURE X.
001091 77  RLDARN                      PICTURE 9(6) COMPUTATIONAL.
001100 77  FORT-KEY                    PICTURE 9(6) COMPUTATIONAL.
001110 77  ACCN-HOLD-5                 PICTURE 9(6) COMPUTATIONAL.
001120 77  KEY-HOLD-8                  PICTURE 9(6) COMPUTATIONAL.
001130 77  KEY-HOLD-10                 PICTURE 9(6) COMPUTATIONAL.
001140 77  KEY-HOLD-A8                 PICTURE 9(6) COMPUTATIONAL.
001150 77  KEY-HOLD-A11                PICTURE 9(6) COMPUTATIONAL.
001160 77  KEY-HOLD-A12                PICTURE 9(6) COMPUTATIONAL.
001170 77  KEY-HOLD-A14                PICTURE 9(6) COMPUTATIONAL.
001180 77  KEY-HOLD-A19                PICTURE 9(6) COMPUTATIONAL.
001190 01  SYM-KEY                     PICTURE X(80).
001200 LINKAGE SECTION.
002010 77  RTLIST                      PICTURE 9(6) COMPUTATIONAL.
002020 01  TITLE-ARG                   PICTURE X(80).
002030 01  ACN-ARG                     PICTURE X(16).
002040 01  DIRA1.
002050         02  ER1                 PICTURE X.
002060         02  DESC-SEEK-KEY       PICTURE X(16).
002070         02  MIDL                PICTURE X(412).
002080         02  NDEX-TO-ACCN        PICTURE 9(6).
002090         02  NINS                PICTURE 9(12).
002100         02  LSTS                PICTURE X(43).
002110 01  DIRA2 REDEFINES DIRA1.
002120         02  ER2                 PICTURE X.
002130         02  TITLE-ISAM          PICTURE X(80).
002140         02  NDEX-TO-TITL1       PICTURE 9(6).
002150         02  FILLER              PICTURE X(243).
```

```
002160 01  DIRA3 REDEFINES DIRA2.
002170      02  TITLE-BUF             PICTURE X(80).
002180      02  NDEX-TO-FRST-REC2     PICTURE 9(6) COMPUTATIONAL.
002190      02  FILLER                PICTURE X(246).
002200 01  DIRA4 REDEFINES DIRA3.
003010      02  NDEX-TO-ACCN-T        PICTURE 9(6) COMPUTATIONAL.
003020      02  NDEX-TO-NEXT-REC2     PICTURE 9(6) COMPUTATIONAL.
003030      02  NDEX-TO-O-TITL        PICTURE 9(6) COMPUTATIONAL.
            02  GARG      PICTURE      9(6)   COMPUTATIONAL.
            02  FILLER      PICTURE    X(474).
003050 01  DIRA5 REDEFINES DIRA4.
003060      02  ACN-BUF               PICTURE X(16).
003070      02  NDEX-TO-TITLE         PICTURE 9(6) COMPUTATIONAL.
003080      02  NDEX-TO-SUBJ          PICTURE 9(6) COMPUTATIONAL.
003090      02  NDEX-TO-AUTH          PICTURE 9(6) COMPUTATIONAL.
003100      02  AUTH-LEVEL            PICTURE 9(6) COMPUTATIONAL.
003110      02  NDEX-TO-NEXT-ACCN     PICTURE 9(6) COMPUTATIONAL.
003120      02  FILLER                PICTURE X(454).
003130 PROCEDURE DIVISION.
003140     ENTER LINKAGE.
003150     ENTRY 'ADDS' USING TITLE-ARG ACN-ARG DIRA1 RTLIST.
003160     ENTER COBOL.
           READY TRACE
003170     MOVE ACN-ARG TO SYM-KEY.
003180     MOVE '0' TO SWIT.
003190     MOVE 3 TO RLDARN.
003200     ENTER LINKAGE.
004010     CALL 'IOENQ' USING RLDARN RTLIST DIRA1 SYM-KEY.
004020     ENTER COBOL.
004030     IF ER1 = '1' GO TO RTRN.
004040     IF NDEX-TO-ACCN = 0 GO TO TWLV.
004050     MOVE NDEX-TO-ACCN TO ACCN-HOLD-5 FORT-KEY.
004060 E3. MOVE 4 TO RLDARN.
004070     ENTER LINKAGE.
004080     CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
004090     ENTER COBOL.
```

```
004100      IF NDEX-TO-SUBJ = 0 GO TO ATE-TEEN.
004110      MOVE NDEX-TO-SUBJ TO FORT-KEY KEY-HOLD-8.
004120 E2.  MOVE 8 TO RLDARN.
004130      ENTER LINKAGE.
004140      CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
004150      ENTER COBOL.
004160      MOVE NDEX-TO-FRST-REC2 TO FORT-KEY KEY-HOLD-10.
004170 E1.  MOVE 9 TO RLDARN.
004180      ENTER LINKAGE.
004190      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
004200      ENTER COBOL.
005010 A.   IF NDEX-TO-ACCN-T = ACCN-HOLD-5 GO TO A2.
005020      MOVE NDEX-TO-NEXT-REC2 TO FORT-KEY KEY-HOLD-10.
005030      GO TO E1.
005040 A2.  IF NDEX-TO-O-TITL = KEY-HOLD-8 GO TO A4.
005050      MOVE NDEX-TO-O-TITL TO FORT-KEY.
005060      GO TO E2.
005070 A4.  MOVE TITLE-ARG TO SYM-KEY.
005080      MOVE 2 TO RLDARN.
005090      ENTER LINKAGE.
005100      CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
005110      ENTER COBOL.
005120      IF ER2 = '1' GO TO A7.
005130      MOVE NDEX-TO-TITL1 TO FORT-KEY KEY-HOLD-A12.
005140      MOVE 8 TO RLDARN.
005150      ENTER LINKAGE.
005160      CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
005170      ENTER COBOL.
005180      MOVE NDEX-TO-FRST-REC2 TO FORT-KEY KEY-HOLD-A14.
005190 A15.
005200      MOVE 9 TO RLDARN.
006010      ENTER LINKAGE.
006020      CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
006030      ENTER COBOL.
006040      IF NDEX-TO-NEXT-REC2 = 0 GO TO A18.
006050      MOVE NDEX-TO-NEXT-REC2 TO FORT-KEY KEY-HOLD-A14.
```

```
006060     GO TO A15.
006070 A18.
006080     MOVE 26 TO RLDARN.
006090     MOVE 6 TO FORT-KEY.
006100     ENTER LINKAGE.
006110     CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
006120     ENTER COBOL.
006130     MOVE FORT-KEY TO KEY-HOLD-A19.
006140     MOVE KEY-HOLD-A14 TO FORT-KEY.
006150     MOVE 38 TO RLDARN.
006160     ENTER LINKAGE.
006170     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
006180     ENTER COBOL.
006190     MOVE KEY-HOLD-A19 TO NDEX-TO-NEXT-REC2.
006200     MOVE KEY-HOLD-A14 TO FORT-KEY.
007010     MOVE 38 TO RLDARN.
007020     ENTER LINKAGE.
007030     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
007040     ENTER COBOL.
007050     MOVE KEY-HOLD-A19 TO KEY-HOLD-A11.
007060     MOVE KEY-HOLD-A12 TO KEY-HOLD-A8.
007070     GO TO B1.
007080 A7.
007090     MOVE 26 TO RLDARN.
007100     MOVE 4 TO FORT-KEY.
007110     ENTER LINKAGE.
007120     CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
007130     ENTER COBOL.
007140     MOVE FORT-KEY TO NDEX-TO-TITL1 KEY-HOLD-A8.
007150     MOVE TITLE-ARG TO TITLE-ISAM SYM-KEY.
007160     MOVE 24 TO RLDARN.
007170     MOVE '3' TO ER2.
007180     ENTER LINKAGE.
007190     CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
007200     ENTER COBOL.
008010     MOVE 26 TO RLDARN.
```

```
008020     MOVE 6 TO FORT-KEY.
008030     ENTER LINKAGE.
008040     CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
008050     ENTER COBOL.
008060     MOVE FORT-KEY TO KEY-HOLD-A11.
008070     MOVE KEY-HOLD-A8 TO FORT-KEY.
008080     MOVE TITLE-ARG TO TITLE-BUF.
008090     MOVE KEY-HOLD-A11 TO NDEX-TO-FRST-REC2.
008100 B.  MOVE 28 TO RLDARN.
008110     ENTER LINKAGE.
008120     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
008130     ENTER COBOL.
008140 B1. MOVE KEY-HOLD-A11 TO FORT-KEY.
008150     MOVE ACCN-HOLD-5 TO NDEX-TO-ACCN-T.
008160     MOVE 0 TO NDEX-TO-NEXT-REC2.
008161     IF SWIT = '1' MOVE KEY-HOLD-A12 TO KEY-HOLD-8.
008170     MOVE KEY-HOLD-8 TO NDEX-TO-O-TITL.
           MOVE ZERO TO GARG.
008180     MOVE 29 TO RLDARN.
008190     ENTER LINKAGE.
008200     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
009010     ENTER COBOL.
009011     IF SWIT = '1' GO TO B8.
009020     MOVE KEY-HOLD-10 TO FORT-KEY.
009030     MOVE 38 TO RLDARN.
009040     ENTER LINKAGE.
009050     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
009060     ENTER COBOL.
009070     MOVE KEY-HOLD-A8 TO NDEX-TO-O-TITL.
009080     MOVE KEY-HOLD-10 TO FORT-KEY.
009090     ENTER LINKAGE.
009100     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
009110     IF SWIT = '1' GO TO B8.
009120 RTRN.
009130     ENTER LINKAGE.
009140     RETURN.
```

```
009150      ENTER COBOL.
009160 B8.  MOVE ACCN-HOLD-5 TO FORT-KEY.
009170      MOVE 33 TO RLDARN.
009180      ENTER LINKAGE.
009190      CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
009200      ENTER COBOL.
010010      MOVE ACCN-HOLD-5 TO FORT-KEY.
010020      MOVE KEY-HOLD-A8 TO NDEX-TO-SUBJ.
010030      ENTER LINKAGE.
010040      CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
010050      ENTER COBOL.
010060      GO TO RTRN.
010070 ATE-TEEN.
010080      MOVE '1' TO SWIT.
010090      GO TO A4.
010100 TWLV.
010110      MOVE 26 TO RLDARN.
010120      MOVE 5 TO FORT-KEY.
010130      ENTER LINKAGE.
010140      CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
010150      ENTER COBOL.
010160      MOVE FORT-KEY TO ACCN-HOLD-5.
010170      MOVE ACN-ARG TO SYM-KEY.
010180      MOVE '1' TO ER1.
010190      MOVE 25 TO RLDARN.
010200      ENTER LINKAGE.
011010      CALL 'IOENQ' USING RLDARN RTLIST DIRA1  SYM-KEY.
011020      ENTER COBOL.
011030      MOVE ACCN-HOLD-5 TO NDEX-TO-ACCN.
011040      MOVE ACN-ARG TO SYM-KEY.
011050      MOVE '2' TO ER1.
011060      ENTER LINKAGE.
011070      CALL 'IOENQ' USING RLDARN RTLIST DIRA1  SYM-KEY.
011080      ENTER COBOL.
011090      MOVE ACCN-HOLD-5 TO FORT-KEY.
011100      MOVE ACN-ARG TO ACN-BUF.
```

```
011110    MOVE 0 TO NDEX-TO-TITLE NDEX-TO-SUBJ NDEX-TO-AUTH
011120    AUTH-LEVEL NDEX-TO-NEXT-ACCN.
011130    MOVE 32 TO RLDARN.
011140    ENTER LINKAGE.
011150    CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
011160    ENTER COBOL.
011170    MOVE ACCN-HOLD-5 TO FORT-KEY.
011180    GO TO E3.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'DROPTITL'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  FORT-KEY                PICTURE 9(6) COMPUTATIONAL.
001100 77  ACCN-HOLD-3             PICTURE 9(6) COMPUTATIONAL.
001110 77  TITL-HOLD-5             PICTURE 9(6) COMPUTATIONAL.
001120 77  TITL1-ONE-BACK          PICTURE 9(6) COMPUTATIONAL.
001130 77  TITL1-HOLD-7            PICTURE 9(6) COMPUTATIONAL.
001140 77  FRST-REC-HOLD-9         PICTURE 9(6) COMPUTATIONAL.
001150 77  FRST-REC2-HOLD-18       PICTURE 9(6) COMPUTATIONAL.
001160 77  BACK-ONE-22             PICTURE 9(6) COMPUTATIONAL.
001170 77  HOLD-THIS-REC-24        PICTURE 9(6) COMPUTATIONAL.
001180 77  HOLD-NEXT2-25           PICTURE 9(6) COMPUTATIONAL.
001190 77  HOLD-OTHER-25           PICTURE 9(6) COMPUTATIONAL.
001200 77  HOLD-REC2-36            PICTURE 9(6) COMPUTATIONAL.
002010 77  HOLD-48                 PICTURE 9(6) COMPUTATIONAL.
002011 77  RLDARN                  PICTURE 9(6) COMPUTATIONAL.
002020 01  SYM-KEY                 PICTURE X(240).
002030 01  HIVAL                   PICTURE X VALUE IS HIGH-VALUE.
002040 LINKAGE SECTION.
002050 77  RTLIST                  PICTURE 9(6) COMPUTATIONAL.
002060 01  TITLE-ARG               PICTURE X(240).
002070 01  ACN-NO                  PICTURE X(16).
002080 01  DIRA1.
002090     02  ER1                 PICTURE X.
002100     02  DESC-SEEK-KEY       PICTURE X(16).
002110     02  MIDL                PICTURE X(412).
002120     02  NDEX-TO-ACCN        PICTURE 9(6).
002130     02  NINS                PICTURE 9(12).
002140     02  LSTS                PICTURE X(43).
002150 01  DIRA2 REDEFINES DIRA1.
```

```
002160     02  ER2                   PICTURE X.
002170     02  TITLE-ISAM            PICTURE X(240).
002180     02  NDEX-TO-TITL1         PICTURE 9(6).
002190     02  FILLER                PICTURE X(243).
002200 01  DIRA3 REDEFINES DIRA2.
003010     02  TITLE-BUF             PICTURE X(240).
003020     02  NDEX-FRST-REC2        PICTURE 9(6) COMPUTATIONAL.
003030     02  FILLER                PICTURE X(246).
003040 01  DIRA4 REDEFINES DIRA3.
003050     02  NDEX-TO-ACCN-T        PICTURE 9(6) COMPUTATIONAL.
003060     02  NDEX-TO-NEXT-REC2     PICTURE 9(6) COMPUTATIONAL.
003070     02  NDEX-TO-OTHER-TITL    PICTURE 9(6) COMPUTATIONAL.
003080     02  FILLER                PICTURE X(478).
003090 01 DIRA5 REDEFINES DIRA4.
003100     02  ACK-NO                PICTURE X(16).
003110     02  NDEX-TO-TITL          PICTURE 9(6) COMPUTATIONAL.
003120     02  FILLER                PICTURE X(470).
003130 PROCEDURE DIVISION.
003140     ENTER LINKAGE.
003150     ENTRY 'DROPT' USING TITLE-ARG ACN-NO DIRA1 RTLIST.
003160     ENTER COBOL.
003170     MOVE ACN-NO TO SYM-KEY DESC-SEEK-KEY.
003180     MOVE 3 TO RLDARN.
003190     ENTER LINKAGE.
003200     CALL 'IOENQ' USING RLDARN RTLIST DIRA1 SYM-KEY.
004010     ENTER COBOL.
004020     MOVE NDEX-TO-ACCN TO FORT-KEY ACCN-HOLD-3.
004030     MOVE 4 TO RLDARN.
004040     ENTER LINKAGE.
004050     CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
004060     ENTER COBOL.
004070     MOVE NDEX-TO-TITL TO FORT-KEY TITL-HOLD-5.
004080     MOVE 6 TO RLDARN.
004090     ENTER LINKAGE.
004100     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
004110     ENTER COBOL.
```

```
004120     MOVE TITL-HOLD-5 TO TITL1-ONE-BACK TITL1-HOLD-7.
004130 ATE.
004140     IF TITLE-BUF = TITLE-ARG GO TO A.
004150     MOVE NDEX-FRST-REC2 TO FORT-KEY FRST-REC-HOLD-9.
004160 TIN.
004170     MOVE 7 TO RLDARN.
004180     ENTER LINKAGE.
004190     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
004200     ENTER COBOL.
005010     IF NDEX-TO-ACCN-T = ACCN-HOLD-3 GO TO FITEEN.
005020     IF NDEX-TO-NEXT-REC2 = 0 GO TO RTRN8.
005030     MOVE NDEX-TO-NEXT-REC2 TO FORT-KEY.
005040     GO TO TIN.
005050 FITEEN.
005060     IF NDEX-TO-OTHER-TITL = TITL-HOLD-5 GO TO RTRN8.
005070     MOVE NDEX-TO-OTHER-TITL TO FORT-KEY.
005080     MOVE TITL1-HOLD-7 TO TITL1-ONE-BACK.
005090     MOVE NDEX-TO-OTHER-TITL TO TITL1-HOLD-7.
005100     MOVE 6 TO RLDARN.
005110     ENTER LINKAGE.
005120     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
005130     ENTER COBOL.
005140     GO TO ATE.
005150 A.
005160     MOVE NDEX-FRST-REC2 TO FORT-KEY FRST-REC2-HOLD-18.
005170     MOVE 0 TO BACK-ONE-22.
005180 NINTEEN.
005190     MOVE 7 TO RLDARN.
005200     ENTER LINKAGE.
006010     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
006020     ENTER COBOL.
006030     IF NDEX-TO-ACCN-T = ACCN-HOLD-3 GO TO TWNTY3.
006040     IF NDEX-TO-NEXT-REC2 = 0 GO TO RTRN8.
006060     MOVE FORT-KEY TO BACK-ONE-22.
006070     MOVE NDEX-TO-NEXT-REC2 TO FORT-KEY.
006080     GO TO NINTEEN.
```

```
006090 TWNTY3.
006100     IF BACK-ONE-22 = 0 GO TO B.
006120     MOVE FORT-KEY TO HOLD-THIS-REC-24.
006130     MOVE NDEX-TO-NEXT-REC2 TO HOLD-NEXT2-25.
006140     MOVE NDEX-TO-OTHER-TITL TO HOLD-OTHER-25.
006150     MOVE BACK-ONE-22 TO FORT-KEY. MOVE 36 TO RLDARN.
006160     ENTER LINKAGE.
006170     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
006180     ENTER COBOL.
006200     MOVE HOLD-NEXT2-25 TO NDEX-TO-NEXT-REC2.
007010     ENTER LINKAGE.
007020     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
007030     ENTER COBOL.
007040     MOVE FRST-REC-HOLD-9 TO FORT-KEY.
007050     MOVE 36 TO RLDARN.
007060     ENTER LINKAGE.
007070     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
007080     ENTER COBOL.
007090     MOVE HOLD-OTHER-25 TO NDEX-TO-OTHER-TITL.
007110     ENTER LINKAGE.
007120     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
007130     ENTER COBOL.
007140 C.
007150     IF TITL1-ONE-BACK = TITL1-HOLD-7 GO TO FOUR-NINE.
007160     MOVE HIVAL TO ACK-NO.
007170     MOVE HOLD-THIS-REC-24 TO FORT-KEY.
007180     MOVE 31 TO RLDARN.
007190     ENTER LINKAGE.
007200     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
008010     ENTER COBOL.
008020 R.  MOVE '0' TO ACK-NO.
008030     ENTER LINKAGE.
008040     RETURN.
008050     ENTER COBOL.
008060 FOUR-NINE.
008070     MOVE HOLD-THIS-REC-24 TO FORT-KEY.
```

```
008080      MOVE 36 TO RLDARN.
008090      ENTER LINKAGE.
008100      CALL 'IDENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
008110      ENTER COBOL.
008130      MOVE HIVAL TO ACK-NO.
008140      MOVE NDEX-TO-OTHER-TITL TO HOLD-48.
008150      ENTER LINKAGE.
008160      CALL 'IDENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
008170      ENTER COBOL.
008180      MOVE ACCN-HOLD-3 TO FORT-KEY.
008190      MOVE 33 TO RLDARN.
008200      ENTER LINKAGE.
009010      CALL 'IDENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
009020      ENTER COBOL.
009040      IF HOLD-48 = NDEX-TO-TITL GO TO FIVE-7.
009050      MOVE HOLD-48 TO NDEX-TO-TITL.
009060 FIVE-9.
009070      ENTER LINKAGE.
009080      CALL 'IDENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
009090      ENTER COBOL.
009100      GO TO R.
009110 FIVE-7.
009120      MOVE 0 TO NDEX-TO-TITL.
009130      GO TO FIVE-9.
009140 B.
009150      IF NDEX-TO-NEXT-REC2 = 0 GO TO THREE-7.
009160      MOVE TITL1-HOLD-7 TO FORT-KEY.
009170      MOVE NDEX-TO-NEXT-REC2 TO HOLD-REC2-36.
009180      MOVE 35 TO RLDARN.
009190      ENTER LINKAGE.
009200      CALL 'IDENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
010010      ENTER COBOL.
010020      MOVE HOLD-REC2-36 TO NDEX-FRST-REC2.
010040      ENTER LINKAGE.
010050      CALL 'IDENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
010060      ENTER COBOL.
```

```
010070 D.
010080     MOVE FRST-REC2-HOLD-18 TO HOLD-THIS-REC-24.
010090     GO TO C.
010100 THREE-7.
010110     MOVE TITLE-ARG TO SYM-KEY.
010120     MOVE '1' TO ER1.
010130     MOVE 23 TO RLDARN.
010140     ENTER LINKAGE.
010150     CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
010160     ENTER COBOL.
010170     MOVE HIVAL TO ER1.
010180     MOVE TITLE-ARG TO SYM-KEY.
010190     ENTER LINKAGE.
010200     CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
011010     ENTER COBOL.
011020     MOVE TITL1-HOLD-7 TO FORT-KEY.
011030     MOVE HIVAL TO TITLE-BUF.
011040     MOVE 30 TO RLDARN.
011050     ENTER LINKAGE.
011060     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
011070     ENTER COBOL.
011080     GO TO D.
011090 RTRN8.
011100     MOVE '8' TO ACK-NO.
011110     ENTER LINKAGE.
011120     RETURN.
011130     ENTER COBOL.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'ADDTIT'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  SWIT                         PICTURE X.
001091 77  RLDARN                       PICTURE 9(6) COMPUTATIONAL.
001100 77  FORT-KEY                     PICTURE 9(6) COMPUTATIONAL.
001110 77  ACCN-HOLD-5                  PICTURE 9(6) COMPUTATIONAL.
001120 77  KEY-HOLD-8                   PICTURE 9(6) COMPUTATIONAL.
001130 77  KEY-HOLD-10                  PICTURE 9(6) COMPUTATIONAL.
001140 77  KEY-HOLD-A8                  PICTURE 9(6) COMPUTATIONAL.
001150 77  KEY-HOLD-A11                 PICTURE 9(6) COMPUTATIONAL.
001160 77  KEY-HOLD-A12                 PICTURE 9(6) COMPUTATIONAL.
001170 77  KEY-HOLD-A14                 PICTURE 9(6) COMPUTATIONAL.
001180 77  KEY-HOLD-A19                 PICTURE 9(6) COMPUTATIONAL.
001190 01  SYM-KEY                      PICTURE X(240).
001200 LINKAGE SECTION.
002010 77  RTLIST                       PICTURE 9(6) COMPUTATIONAL.
002020 01  TITLE-ARG                    PICTURE X(240).
002030 01  ACN-ARG                      PICTURE X(16).
002040 01  DIRA1.
002050         02  ER1                  PICTURE X.
002060         02  DESC-SEEK-KEY        PICTURE X(16).
002070         02  MIDL                 PICTURE X(412).
002080         02  NDEX-TO-ACCN         PICTURE 9(6).
002090         02  NINS                 PICTURE 9(12).
002100         02  LSTS                 PICTURE X(43).
002110 01  DIRA2 REDEFINES DIRA1.
002120         02  ER2                  PICTURE X.
002130         02  TITLE-ISAM           PICTURE X(240).
002140         02  NDEX-TO-TITL1        PICTURE 9(6).
002150         02  FILLER               PICTURE X(243).
```

```
002160 01   DIRA3 REDEFINES DIRA2.
002170        02  TITLE-BUF             PICTURE X(240).
002180        02  NDEX-TO-FRST-REC2     PICTURE 9(6) COMPUTATIONAL.
002190        02  FILLER                PICTURE X(246).
002200 01   DIRA4 REDEFINES DIRA3.
003010        02  NDEX-TO-ACCN-T        PICTURE 9(6) COMPUTATIONAL.
003020        02  NDEX-TO-NEXT-REC2     PICTURE 9(6) COMPUTATIONAL.
003030        02  NDEX-TO-O-TITL        PICTURE 9(6) COMPUTATIONAL.
003040        02  FILLER                PICTURE X(478).
003050 01   DIRA5 REDEFINES DIRA4.
003060        02  ACN-BUF               PICTURE X(16).
003070        02  NDEX-TO-TITLE         PICTURE 9(6) COMPUTATIONAL.
003080        02  NDEX-TO-SUBJ          PICTURE 9(6) COMPUTATIONAL.
003090        02  NDEX-TO-AUTH          PICTURE 9(6) COMPUTATIONAL.
003100        02  AUTH-LEVEL            PICTURE 9(6) COMPUTATIONAL.
003110        02  NDEX-TO-NEXT-ACCN     PICTURE 9(6) COMPUTATIONAL.
003120        02  FILLER                PICTURE X(454).
003130 PROCEDURE DIVISION.
003140      ENTER LINKAGE.
003150      ENTRY 'ADDT' USING TITLE-ARG ACN-ARG DIRA1 RTLIST.
003160      ENTER COBOL.
003170      MOVE ACN-ARG TO SYM-KEY.
003180      MOVE '0' TO SWIT.
003190      MOVE 3 TO RLDARN.
003200      ENTER LINKAGE.
004010      CALL 'IOENQ' USING RLDARN RTLIST DIRA1 SYM-KEY.
004020      ENTER COBOL.
004030      IF ER1 = '1' GO TO RTRN.
004040      IF NDEX-TO-ACCN = 0 GO TO TWLV.
004050      MOVE NDEX-TO-ACCN TO ACCN-HOLD-5 FORT-KEY.
004060 E3.  MOVE 4 TO RLDARN.
004070      ENTER LINKAGE.
004080      CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
004090      ENTER COBOL.
004100      IF NDEX-TO-TITLE = 0 GO TO ATE-TEEN.
004110      MOVE NDEX-TO-TITLE TO FORT-KEY KEY-HOLD-8.
```

```
004120 E2. MOVE 6 TO RLDARN.
004130     ENTER LINKAGE.
004140     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
004150     ENTER COBOL.
004160     MOVE NDEX-TO-FRST-REC2 TO FORT-KEY KEY-HOLD-10.
004170 E1. MOVE 7 TO RLDARN.
004180     ENTER LINKAGE.
004190     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
004200     ENTER COBOL.
005010 A.  IF NDEX-TO-ACCN-T = ACCN-HOLD-5 GO TO A2.
005020     MOVE NDEX-TO-NEXT-REC2 TO FORT-KEY KEY-HOLD-10.
005030     GO TO E1.
005040 A2. IF NDEX-TO-O-TITL = KEY-HOLD-8 GO TO A4.
005050     MOVE NDEX-TO-O-TITL TO FORT-KEY.
005060     GO TO E2.
005070 A4. MOVE TITLE-ARG TO SYM-KEY.
005080     MOVE 1 TO RLDARN.
005090     ENTER LINKAGE.
005100     CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
005110     ENTER COBOL.
005120     IF ER2 = '1' GO TO A7.
005130     MOVE NDEX-TO-TITL1 TO FORT-KEY KEY-HOLD-A12.
005140     MOVE 6 TO RLDARN.
005150     ENTER LINKAGE.
005160     CALL 'IOENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
005170     ENTER COBOL.
005180     MOVE NDEX-TO-FRST-REC2 TO FORT-KEY KEY-HOLD-A14.
005190 A15.
005200     MOVE 7 TO RLDARN.
006010     ENTER LINKAGE.
006020     CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
006030     ENTER COBOL.
006040     IF NDEX-TO-NEXT-REC2 = 0 GO TO A18.
006050     MOVE NDEX-TO-NEXT-REC2 TO FORT-KEY KEY-HOLD-A14.
006060     GO TO A15.
006070 A18.
```

```
006080        MOVE 26 TO RLDARN.
006090        MOVE 3 TO FORT-KEY.
006100        ENTER LINKAGE.
006110        CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
006120        ENTER COBOL.
006130        MOVE FORT-KEY TO KEY-HOLD-A19.
006140        MOVE KEY-HOLD-A14 TO FORT-KEY.
006150        MOVE 36 TO RLDARN.
006160        ENTER LINKAGE.
006170        CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
006180        ENTER COBOL.
006190        MOVE KEY-HOLD-A19 TO NDEX-TO-NEXT-REC2.
006200        MOVE KEY-HOLD-A14 TO FORT-KEY.
007010        MOVE 36 TO RLDARN.
007020        ENTER LINKAGE.
007030        CALL 'IOENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
007040        ENTER COBOL.
007050        MOVE KEY-HOLD-A19 TO KEY-HOLD-A11.
007060        MOVE KEY-HOLD-A12 TO KEY-HOLD-A8.
007070        GO TO B1.
007080    A7.
007090        MOVE 26 TO RLDARN.
007100        MOVE 2 TO FORT-KEY.
007110        ENTER LINKAGE.
007120        CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
007130        ENTER COBOL.
007140        MOVE FORT-KEY TO NDEX-TO-TITL1 KEY-HOLD-A8.
007150        MOVE TITLE-ARG TO TITLE-ISAM SYM-KEY.
007160        MOVE 23 TO RLDARN.
007170        MOVE '3' TO ER2.
007180        ENTER LINKAGE.
007190        CALL 'IOENQ' USING RLDARN RTLIST DIRA2 SYM-KEY.
007200        ENTER COBOL.
008010        MOVE 26 TO RLDARN.
008020        MOVE 3 TO FORT-KEY.
008030        ENTER LINKAGE.
```

```
008040      CALL 'IDENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
008050      ENTER COBOL.
008060      MOVE FORT-KEY TO KEY-HOLD-A11.
008070      MOVE KEY-HOLD-A8 TO FORT-KEY.
008080      MOVE TITLE-ARG TO TITLE-BUF.
008090      MOVE KEY-HOLD-A11 TO NDEX-TO-FRST-REC2.
008100 B.   MOVE 30 TO RLDARN.
008110      ENTER LINKAGE.
008120      CALL 'IDENQ' USING RLDARN RTLIST DIRA3 FORT-KEY.
008130      ENTER COBOL.
008140 B1.  MOVE KEY-HOLD-A11 TO FORT-KEY.
008150      MOVE ACCN-HOLD-5 TO NDEX-TO-ACCN-T.
008160      MOVE 0 TO NDEX-TO-NEXT-REC2.
008161      IF SWIT = '1' MOVE KEY-HOLD-A12 TO KEY-HOLD-8.
008170      MOVE KEY-HOLD-8 TO NDEX-TO-O-TITL.
008180      MOVE 31 TO RLDARN.
008190      ENTER LINKAGE.
008200      CALL 'IDENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
009010      ENTER COBOL.
009011      IF SWIT = '1' GO TO B8.
009020      MOVE KEY-HOLD-10 TO FORT-KEY.
009030      MOVE 36 TO RLDARN.
009040      ENTER LINKAGE.
009050      CALL 'IDENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
009060      ENTER COBOL.
009070      MOVE KEY-HOLD-A8 TO NDEX-TO-O-TITL.
009080      MOVE KEY-HOLD-10 TO FORT-KEY.
009090      ENTER LINKAGE.
009100      CALL 'IDENQ' USING RLDARN RTLIST DIRA4 FORT-KEY.
009110      IF SWIT = '1' GO TO B8.
009120 RTRN.
009130      ENTER LINKAGE.
009140      RETURN.
009150      ENTER COBOL.
009160 B8.  MOVE ACCN-HOLD-5 TO FORT-KEY.
009170      MOVE 33 TO RLDARN.
```

```
009180       ENTER LINKAGE.
009190       CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
009200       ENTER COBOL.
010010       MOVE ACCN-HOLD-5 TO FORT-KEY.
010020       MOVE KEY-HOLD-A8 TO NDEX-TO-TITLE.
010030       ENTER LINKAGE.
010040       CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
010050       ENTER COBOL.
010060       GO TO RTRN.
010070 ATE-TEEN.
010080       MOVE '1' TO SWIT.
010090       GO TO A4.
010100 TWLV.
010110       MOVE 26 TO RLDARN.
010120       MOVE 5 TO FORT-KEY.
010130       ENTER LINKAGE.
010140       CALL 'IOENQ' USING RLDARN RTLIST DIRA5 FORT-KEY.
010150       ENTER COBOL.
010160       MOVE FORT-KEY TO ACCN-HOLD-5.
010170       MOVE ACN-ARG TO SYM-KEY.
010180       MOVE '1' TO ER1.
010190       MOVE 25 TO RLDARN.
010200       ENTER LINKAGE.
011010       CALL 'IOENQ' USING RLDARN RTLIST DIRA1 SYM-KEY.
011020       ENTER COBOL.
011030       MOVE ACCN-HOLD-5 TO NDEX-TO-ACCN.
011040       MOVE ACN-ARG TO SYM-KEY.
011050       MOVE '2' TO ER1.
011060       ENTER LINKAGE.
011070       CALL 'IOENQ' USING RLDARN RTLIST DIRA1 SYM-KEY.
011080       ENTER COBOL.
011090       MOVE ACCN-HOLD-5 TO FORT-KEY.
011100       MOVE ACN-ARG TO ACN-BUF.
011110       MOVE 0 TO NDEX-TO-TITLE NDEX-TO-SUBJ NDEX-TO-AUTH
011120       AUTH-LEVEL NDEX-TO-NEXT-ACCN.
011130       MOVE 32 TO RLDARN.
```

```
011140      ENTER LINKAGE.
011150      CALL 'IDENQ' USING RLDARN RTLIST DIRAS FORT-KEY.
011160      ENTER COBOL.
011170      MOVE ACCN-HOLD-5 TO FORT-KEY.
011180      GO TO E3.
```

```
001010 IDENTIFICATION DIVISION.
   020 PROGRAM-ID. 'M1'.
   030 ENVIRONMENT DIVISION.
   040 CONFIGURATION SECTION.
   050 SOURCE-COMPUTER. IBM-360.
   060 OBJECT-COMPUTER. IBM-360.
   070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
002010 LINKAGE SECTION.
   020 77  RTLIST                      PICTURE 9(6) COMPUTATIONAL.
002021 77  WHICH-ONE                   PICTURE 9(6) COMPUTATIONAL.
   030 77  SWIT                        PICTURE 9(8) COMPUTATIONAL.
   040 01  REG8.
   050     02  FILLER                  PICTURE X(4).
   060     02  RORW                    PICTURE 9(6) COMPUTATIONAL.
           02  LENG                    PICTURE 9(6) COMPUTATIONAL.
   070 01  MESAGE.
   080     02  ID-NO                   PICTURE X(10).
   090     02  FILLER                  PICTURE X(470).
   100 01  SCOPE.
002101     02  FILLER                  PICTURE X(4).
   110     02  L1.
   120         03  USER-ID             PICTURE X(10).
   130         03  FILLER              PICTURE X(110).
   140     02  L2                      PICTURE X(120).
   150     02  L3                      PICTURE X(120).
   160     02  L4                      PICTURE X(120).
   170 01  DIRA.
002180     02  FILLER                  PICTURE X(428).
003010 PROCEDURE DIVISION.
   020     ENTER LINKAGE.
   030     ENTRY 'MOD1' USING REG8 RTLIST MESAGE SCOPE DIRA SWIT.
003031     ENTER COBOL.
   040     MOVE USER-ID TO ID-NO.
003041     MOVE SPACES TO SCOPE.
   050     MOVE 'YOU MAY SEARCH BY AUTHOR,TITLE,SUBJECT  OR ANY COMBINAT
   060-    'ION THEREOF. ENTER YOUR  CATEGORY FROM ONE OF THE FOLLOWING
003070-    '      ' TO L1.
003071     MOVE 'A,T,S,ATS,AS,TS OR AT' TO L2.
003072     MOVE 160 TO WHICH-ONE.
003073     ENTER LINKAGE.
           CALL 'GOSYM' USING L1 WHICH-ONE.
003075     ENTER COBOL.
003080     MOVE 8 TO RORW. MOVE 160 TO LENG.
003090     MOVE 33554433 TO SWIT.
   100     ENTER LINKAGE.
   110     RETURN.
003120     ENTER COBOL.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'M2'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  SUB                      PICTURE 9(3) VALUE 1.
001091 77  WHICH-ONE                PICTURE 9(6) COMPUTATIONAL.
002010 LINKAGE SECTION.
002020 77  RTLIST                   PICTURE 9(6) COMPUTATIONAL.
002030 77  SWIT                     PICTURE 9(8) COMPUTATIONAL.
002040 01  REG8.
002050     02  FILLER               PICTURE X(4).
002060     02  RORW                 PICTURE 9(6) COMPUTATIONAL.
002070     02  LENG                 PICTURE 9(6) COMPUTATIONAL.
002080 01  MESAGE.
002090     02  ID-NO                PICTURE X(10).
002100     02  FILLER               PICTURE X(470).
002110 01  SCOPE.
002111     02  LENGH                PICTURE 9(6) COMPUTATIONAL.
002112     02  SCOPER.
002120         03  L1.
002130             04  REPLY        PICTURE X OCCURS 40 TIMES.
002140             04 FILLER        PICTURE X(80).
002150         03  L3               PICTURE X(360).
        02  SCOPE-OUT REDEFINES SCOPER.
            03  FIRST-120.
                04  L1-1            PICTURE X(40).
                04  L2-1            PICTURE X(40).
                04  L3-1            PICTURE X(40).
            03  SECOND-120.
                04  L1-2            PICTURE X(40).
                04  L2-2            PICTURE X(40).
                04  L3-2            PICTURE X(40).
```

```
           03  THIRD-120.
               04  L1-3              PICTURE X(40).
               04  L2-3              PICTURE X(40).
               04  L3-3              PICTURE X(40).
           03  FOURTH-120.
               04  L1-4              PICTURE X(40).
               04  L2-4              PICTURE X(40).
               04  L3-4              PICTURE X(40).
002160 01  DIRA.
002170     02  FILLER                PICTURE X(428).
003010 PROCEDURE DIVISION.
003020     ENTER LINKAGE.
003030     ENTRY 'MOD2' USING REG8 RTLIST MESAGE SCOPE DIRA SWIT.
003040     ENTER COBOL.
003050     IF LENGH   2 GO TO TEST3.
003060     IF LENGH ) 2 GO TO TEST1.
003070     IF REPLY (SUB) = 'A' GO TO TEST-AT.
003080     IF REPLY (SUB) = 'T' GO TO TEST-S.
003090     GO TO ERR.
003100 TEST-S.
003110     ADD 1 TO SUB.
003120     IF REPLY (SUB) = 'S' GO TO TITL-SUB.
003130     GO TO ERR.
003140 TEST-AT.
003150     ADD 1 TO SUB.
003160     IF REPLY (SUB) = 'S' GO TO AUTH-SUB.
003170     IF REPLY (SUB) = 'T' GO TO AUTH-TITL.
003180     GO TO ERR.
003190 TEST3.
003200     IF REPLY (SUB) IS NOT EQUAL TO 'A' GO TO ERR.
003210     ADD 1 TO SUB.
003220     IF REPLY (SUB) IS NOT EQUAL TO 'T' GO TO ERR.
003230     ADD 1 TO SUB.
003240     IF REPLY (SUB) IS NOT EQUAL TO 'S' GO TO ERR.
003250     GO TO AUTH-TITL-SUB.
004010 TEST1.
```

```
004020     IF REPLY (SUB) = 'A' GO TO AUTH.
004030     IF REPLY (SUB) = 'T' GO TO TITL.
004040     IF REPLY (SUB) = 'S' GO TO SUBJ.
004050 AUTH-SUB.
004051     MOVE 33554438 TO SWIT.
004060 A.  MOVE 'PLEASE SUBMIT YOUR REQUESTED AUTHOR,    LAST NAME FIRST
004070-    'AND NOT MORE THAN 213    CHARACTERS.' TO SCOPE-OUT.
004080     MOVE 120 TO WHICH-ONE.
004081     ENTER LINKAGE.
004082     CALL 'GOSYM' USING SCOPE-OUT WHICH-ONE.
004083     ENTER COBOL.
004090     MOVE 8 TO RORW.
004100     MOVE 120 TO LENG.
004120 RTURN.
004130     ENTER LINKAGE.
004140     RETURN.
004150     ENTER COBOL.
004160 TITL-SUB.
004161     MOVE 33554439 TO SWIT.
004170 T.  MOVE 'PLEASE SUBMIT YOUR REQUESTED TITLE,     NOT MORE THAN 2
004180-    '40 CHARACTERS.' TO SCOPE-OUT.
004181     MOVE 80 TO WHICH-ONE.
004182     ENTER LINKAGE.
004183     CALL 'GOSYM' USING SCOPE-OUT WHICH-ONE.
004184     ENTER COBOL.
004190     MOVE 8 TO RORW.
004200     MOVE 80 TO LENG.
004220     GO TO RTURN.
004230 AUTH-TITL.
004240     MOVE 'PLEASE SUBMIT YOUR REQUESTED AUTHOR,    LAST NAME FIRST
004250-    'AND NOT MORE THAN 213    CHARACTERS.' TO SCOPE-OUT.
004251     MOVE 120 TO WHICH-ONE.
004252     ENTER LINKAGE.
004253     CALL 'GOSYM' USING SCOPE-OUT WHICH-ONE.
004254     ENTER COBOL.
005020     MOVE 8 TO RORW.
```

```
005030      MOVE 120 TO LENG.
005040      MOVE 33554440 TO SWIT.
005050      GO TO RTURN.
005060 AUTH-TITL-SUB.
005070      MOVE 33554437 TO SWIT.
005080      GO TO A.
005090 AUTH.
005100      MOVE 33554434 TO SWIT.
005110      GO TO A.
005120 TITL.
005130      MOVE 33554435 TO SWIT.
005140      GO TO T.
005150 SUBJ.
005160      MOVE 'PLEASE SUBMIT YOUR REQUESTED SUBJECT,   NOT MORE THAN 8
005170-     '0 CHARACTERS.' TO SCOPE-OUT.
005171      MOVE 80 TO WHICH-ONE.
005172      ENTER LINKAGE.
005173      CALL 'GOSYM' USING SCOPE-OUT WHICH-ONE.
005174      ENTER COBOL.
005180      MOVE 8 TO RORW.
005190      MOVE 80 TO LENG.
005200      MOVE 33554436 TO SWIT.
005210      GO TO RTURN.
005220 ERR.
005230      MOVE 'YOUR RESPONSE IS INCORRECT-TRY AGAIN.' TO SCOPE-OUT.
005231      MOVE 40 TO WHICH-ONE.
005232      ENTER LINKAGE.
005233      CALL 'GOSYM' USING SCOPE-OUT WHICH-ONE.
005234      ENTER COBOL.
005240      MOVE 8 TO RORW.
005250      MOVE 40 TO LENG.
006010      MOVE 33554433 TO SWIT.
006020      GO TO RTURN.
```

```
001010 IDENTIFICATION DIVISION.
001020 PROGRAM-ID. 'M3'.
001030 ENVIRONMENT DIVISION.
001040 CONFIGURATION SECTION.
001050 SOURCE-COMPUTER. IBM-360.
001060 OBJECT-COMPUTER. IBM-360.
001070 DATA DIVISION.
001080 WORKING-STORAGE SECTION.
001090 77  SUB                      PICTURE 9(3) VALUE 1.
001091 77  RLDARN                   PICTURE 9(6) COMPUTATIONAL.
001092 77  FR-NM-HOLD               PICTURE 9(6) COMPUTATIONAL.
001093 77  FORT-KEY                 PICTURE 9(6) COMPUTATIONAL.
001094 77  AUT-INIT-SWIT            PICTURE 9 VALUE ZERO.
001095 77  SUB-HOLD                 PICTURE 9(3).
001096 77  SUB1                     PICTURE 9(3) VALUE 1.
001100 01  AUT-KEY.
001110         02  AUTH-KEY         PICTURE X OCCURS 213 TIMES.
001111 01  AUTH-SAVE                PICTURE X(240).
001112 01  AUTH-IN-SAVE.
001113         02  AUTH-INN-SAVE    PICTURE X OCCURS 213 TIMES.
002010 LINKAGE SECTION.
002020 77  RTLIST                   PICTURE 9(6) COMPUTATIONAL.
002030 77  SWIT                     PICTURE 9(8) COMPUTATIONAL.
002031 77  WHICH-ONE                PICTURE 9(6) COMPUTATIONAL.
002040 01  REG8.
002050         02  FILLER           PICTURE X(4).
002060         02  RORW             PICTURE 9(6) COMPUTATIONAL.
002070         02  LENG             PICTURE 9(6) COMPUTATIONAL.
002080 01  MESAGE.
002090         02  ID-NO            PICTURE X(10).
002100         02  RTRN-INDIC       PICTURE 9(3) COMPUTATIONAL.
002110         02  COUNT-SAVE       PICTURE 9(6) COMPUTATIONAL.
002120         02  ALSO-COUNT       PICTURE 9(6) COMPUTATIONAL.
002130         02  POINTER1.
002131             03  PTARAY1 PICTURE 9(6) COMPUTATIONAL OCCURS 114 TIMES.
002132         02  LINK.
```

```
002133          03  BLK-LEN         PICTURE 9(4) COMPUTATIONAL.
002134          03  LNK-NO          PICTURE 9(4) COMPUTATIONAL.
002140 01   SCOPE.
002150      02  MES-LENG            PICTURE 9(5) COMPUTATIONAL.
002151      02  AUTH-INN.
002160          03  AUTH-IN         PICTURE X OCCURS 213 TIMES.
002170      02  FILLER              PICTURE X(267).
002180 01   SCOPE-OUT REDEFINES SCOPE.
002190      02  MES-LEN             PICTURE 9(5) COMPUTATIONAL.
002200      02  TOTAL-SCOPE.
002201          03  FIRST-120.
002202              04  L1-1.
002203                  05  L1A     PICTURE X(10).
002204                  05  CNT-OUT PICTURE 9(5).
002205                  05  L1B     PICTURE X(25).
002206              04  L2-1        PICTURE X(40).
002207              04  L3-1        PICTURE X(40).
002208          03  SECOND-120.
002209              04  L1-2        PICTURE X(40).
002210              04  L2-2        PICTURE X(40).
002211              04  L3-2        PICTURE X(40).
002212          03  THIRD-120.
002213              04  L1-3        PICTURE X(40).
002214              04  L2-3        PICTURE X(40).
002215              04  L3-3        PICTURE X(40).
002216          03  FOURTH-120.
002217              04  L1-4        PICTURE X(40).
002218              04  L2-4        PICTURE X(40).
002219              04  L3-4        PICTURE X(40).
002230 01   DIRA.
002240      02  ERR-IND             PICTURE X.
002250      02  AUT-LST-NM          PICTURE X(213).
003010      02  INDEX-FRST-NM       PICTURE 9(6).
003020      02  FILLER              PICTURE X(210).
003030 01   DIRA1 REDEFINES DIRA.
003040      02  AUT-HOR             PICTURE X(213).
```

```
003050      02  NDEX-NEXT-AUT        PICTURE 9(5) COMPUTATIONAL.
003060      02  NDEX-ACCN-A          PICTURE 9(5) COMPUTATIONAL.
003070      02  SEE-INDX             PICTURE 9(5) COMPUTATIONAL.
003080      02  YOB                  PICTURE X(8).
003090      02  FILLER               PICTURE X(195).
003100 01   DIRA2 REDEFINES DIRA1.
003110      02  ACCES-NO             PICTURE X(16).
003120      02  NDEX-T-TITL          PICTURE 9(6) COMPUTATIONAL.
003130      02  NDEX-T-SUBJ          PICTURE 9(6) COMPUTATIONAL.
003140      02  NDEX-T-AUT           PICTURE 9(6) COMPUTATIONAL.
003150      02  NDEX-T-O-AUT         PICTURE 9(6) COMPUTATIONAL.
003160      02  A-LEV                PICTURE 9(6) COMPUTATIONAL.
003170      02  NDEX-T-N-ACCN        PICTURE 9(6) COMPUTATIONAL.
003180      02  FILLER               PICTURE X(388).
004010 PROCEDURE DIVISION.
004020      ENTER LINKAGE.
004030      ENTRY 'MOD3' USING REG8 RTLIST MESAGE SCOPE DIRA SWIT.
004040      ENTER COBOL.
004041      MOVE 1 TO BLK-LEN.
004042      MOVE 0 TO LNK-NO.
004043      MOVE 0 TO RTRN-INDIC.
004050      MOVE SPACES TO AUT-KEY.
004060 A.   IF AUTH-IN (SUB) = ',' GO TO END-NM.
004070      MOVE AUTH-IN (SUB) TO AUTH-KEY (SUB).
004080      ADD 1 TO SUB.
004090      IF SUB   213 GO TO END-NM.
004100      GO TO A.
004110 END-NM.
004120      MOVE 0 TO RLDARN.
004130      MOVE SPACES TO DIRA.
004140      ENTER LINKAGE.
004150      CALL 'IDENQ' USING RLDARN RTLIST DIRA AUT-KEY.
004160      ENTER COBOL.
004170      IF ERR-IND = '1' GO TO MOD3B.
004180      MOVE 5 TO RLDARN.
004190      MOVE INDEX-FRST-NM TO FORT-KEY FR-NM-HOLD.
```

```
004200 B.   MOVE SPACES TO DIRA.
004210      ENTER LINKAGE.
004220      CALL 'IOENQ' USING RLDARN RTLIST DIRA FORT-KEY.
004230      ENTER COBOL.
004240      IF AUT-HOR = AUTH-INN MOVE 1 TO SUB COUNT-SAVE GO TO MOD3C.
004250      IF NDEX-NEXT-AUT = 0 GO TO END-NM1.
005010      MOVE NDEX-NEXT-AUT TO FORT-KEY.
005020      GO TO B.
005030 END-NM1.
005040      IF AUT-INIT-SWIT = 1 GO TO MOD3E.
005050 MOD3A.
005060      MOVE SUB TO SUB-HOLD. MOVE 1 TO SUB.
005070      MOVE SPACES TO AUT-KEY.
005080 C.   MOVE AUTH-IN (SUB) TO AUTH-KEY (SUB).
005090      ADD 1 TO SUB.
005100      IF SUB   SUB-HOLD GO TO D.
005110      GO TO C.
005120 D.
005130      IF AUTH-IN (SUB) = ' ' MOVE AUTH-IN (SUB) TO AUTH-KEY (SUB)
005140      GO TO E.
005150      MOVE SUB TO SUB-HOLD. ADD 1 TO SUB-HOLD.
005160      MOVE AUTH-IN (SUB) TO AUTH-KEY (SUB-HOLD).
005170 D1.  ADD 1 TO SUB-HOLD.
005180      MOVE '.' TO AUTH-KEY (SUB-HOLD).
005190      ADD 2 TO SUB-HOLD.
005200      ADD 1 TO SUB.
005210 D2.  IF SUB   213 GO TO SET-UP.
005220      IF AUTH-IN (SUB) IS NOT EQUAL TO ' ' ADD 1 TO SUB GO TO D2.
005230      GO TO E2.
006010 E.   MOVE SUB TO SUB-HOLD. ADD 1 TO SUB-HOLD.
006020      ADD 1 TO SUB.
006030 E2.  IF AUTH-IN (SUB) = ' ' GO TO E1.
006040      MOVE AUTH-IN (SUB) TO AUTH-KEY (SUB-HOLD).
006050      GO TO D1.
006060 E1.
006070      ADD 1 TO SUB.
```

```
006080      IF SUB   213 GO TO SET-UP.
006090      GO TO E2.
006100 SET-UP.
006110      MOVE 1 TO AUT-INIT-SWIT.
006120      MOVE FR-NM-HOLD TO FORT-KEY.
006130      MOVE 5 TO RLDARN.
006140      MOVE AUTH-INN TO AUTH-SAVE.
006150      MOVE AUT-KEY TO AUTH-INN.
006160      GO TO B.
006170 MOD3B.
006171      MOVE SPACES TO TOTAL-SCOPE.
006180      MOVE 'THERE IS NO INFORMATION ON FILE FOR YOURREQUEST, TRY AG
006190-     'AIN.' TO TOTAL-SCOPE.
006191      MOVE 80 TO WHICH-ONE.
006192      ENTER LINKAGE.
006193      CALL 'GOSYM' USING TOTAL-SCOPE WHICH-ONE.
006194      ENTER COBOL.
006200      MOVE 8 TO RORW.
006210      MOVE 80 TO LENG.
006220      MOVE 33554434 TO SWIT.
006230 RT.  ENTER LINKAGE.
006240      RETURN.
006250      ENTER COBOL.
007010 MOD3C.
007011      MOVE NDEX-ACCN-A TO PTARAY1 (SUB).
007030      MOVE 4 TO RLDARN.
007040      MOVE NDEX-ACCN-A TO FORT-KEY.
007050      ENTER LINKAGE.
007060      CALL 'IOENQ' USING RLDARN RTLIST DIRA FORT-KEY.
007070      ENTER COBOL.
007080      IF NDEX-T-N-ACCN = 0 GO TO MOD3D.
007081      ADD 1 TO COUNT-SAVE.
007082      ADD 1 TO SUB.
007083      ADD 1 TO BLK-LEN.
007084      IF SUB   114 GO TO CALL-WRT.
007090 F.   MOVE NDEX-T-N-ACCN TO NDEX-ACCN-A.
```

```
007100      GO TO MOD3C.
007110 MOD3D.
007120      IF COUNT-SAVE   1 GO TO MOD3D1.
007130      MOVE 16 TO RORW.
007140      MOVE 67108873 TO SWIT.
007150      GO TO RT.
007151 CALL-WRT.
007152      MOVE 17 TO RLDARN.
007153      ENTER LINKAGE.
007154      CALL 'IDENQ' USING RLDARN RTLIST POINTER1 LINK.
007155      ENTER COBOL.
007156      MOVE 1 TO SUB BLK-LEN.
007157      GO TO F.
007160 MOD3D1.
007170      MOVE SPACES TO TOTAL-SCOPE.
007180      MOVE 'THERE ARE ' TO L1A.
007190      MOVE COUNT-SAVE TO CNT-OUT.
007200      MOVE ' CITATIONS RELATIVE TO  ' TO L1B.
007210      MOVE 'YOUR REQUEST. WOULD YOU CARE TO FURTHER ' TO L2-1.
007220      MOVE 'DEFINE YOUR SEARCH WITH THE FOLLOWING   ' TO L3-1.
007230      MOVE 'OPTIONS..............................BOOKS ONLY, JOU
007231-     'RNALS ONLY, DATE OF      PUBLICATION YY TO YY, LANGUAGE.'
007232      TO SECOND-120.
007233      MOVE 240 TO WHICH-ONE.
007234      ENTER LINKAGE.
007235      CALL 'GOSYM' USING TOTAL-SCOPE WHICH-ONE.
007236      ENTER COBOL.
007240      MOVE 8 TO RORW.
007250      MOVE 240 TO LENG.
008010      MOVE 33554442 TO SWIT.
008020      GO TO RT.
008030 MOD3E.
008040      MOVE AUTH-SAVE TO AUTH-INN.
008050      MOVE FR-NM-HOLD TO FORT-KEY.
008060 MOD3E-3.
008070      MOVE SPACES TO AUT-KEY.
```

```
008080     MOVE 1 TO SUB SUB1. MOVE 5 TO RLDARN. MOVE SPACES TO DIRA.
008090     ENTER LINKAGE.
008100     CALL 'IDENQ' USING RLDARN RTLIST DIRA FORT-KEY.
008110     ENTER COBOL.
008120     MOVE AUT-HOR TO AUTH-IN-SAVE.
008130 MOD3E-1.
008140     MOVE AUTH-INN-SAVE (SUB) TO AUTH-KEY (SUB1).
008150     IF AUTH-INN-SAVE (SUB) = ',' GO TO COMA-FOUND.
008160     ADD 1 TO SUB. ADD 1 TO SUB1.
008170     IF SUB   213 GO TO MOD3E-END.
008180     GO TO MOD3E-1.
008190 COMA-FOUND.
008200     ADD 2 TO SUB. ADD 2 TO SUB1.
008210     MOVE AUTH-INN-SAVE (SUB) TO AUTH-KEY (SUB1).
008220     ADD 1 TO SUB1. MOVE '.' TO AUTH-KEY (SUB1).
008230 MOD3E-2.
008240     IF SUB   213 GO TO MOD3E-END.
009010     ADD 1 TO SUB.
009020     IF AUTH-INN-SAVE (SUB) IS NOT EQUAL TO ' ' GO TO MOD3E-2.
009030     ADD 1 TO SUB.
009040     IF AUTH-INN-SAVE (SUB) = ' ' GO TO MOD3E-END.
009050     SUBTRACT 2 FROM SUB. GO TO COMA-FOUND.
009060 MOD3E-END.
009070     IF AUTH-INN = AUT-KEY MOVE 1 TO SUB COUNT-SAVE GO TO MOD3C.
009050     IF NDEX-NEXT-AUT = 0 GO TO MOD3B.
009090     MOVE NDEX-NEXT-AUT TO FORT-KEY GO TO MOD3E-3.
```

```
      PROGRAM-ID.   'M4'.
      ENVIRONMENT DIVISION.
      CONFIGURATION SECTION.
      SOURCE-COMPUTER.   IBM-360.
      OBJECT-COMPUTER.   IBM-360.
      DATA DIVISION.
      WORKING-STORAGE SECTION.
      01   SAV   PICTURE   9(6)    COMPUTATIONAL.
      01   RLDARN     PICTURE  9(6)  COMPUTATIONAL.
      01   SYMBO-KEY     PICTURE  X(240).
      01   REL-KEY           PICTURE      9(6)    COMPUTATIONAL.
      01   GOSYLENT    PICTURE   9(6)  COMPUTATIONAL.
      01   SCOPE.
           02  FILLER    PICTURE   XXXX.
           02  L1     PICTURE   X(280).
           02  L2     PICTURE  X(120)   VALUE
           'THERE ARE FORMAT ERRORS IN THE ABOVE    DATA. PLEASE RE-ENTE
      -    'R TITLE-NO MORE THAN 240 CHARACTERS.'.
      LINKAGE SECTION.
      01   SCOP.
           02 LEGTH     PICTURE     X(4).
           02  LL1     PICTURE      X(240).
           02  FILLER      PICTURE      X(240).
      01   DIRA.
           02  HIER-COD    PICTURE X.
           02 FILLER      PICTURE     X(240).
           02 INDX-TITL1      PICTURE      9(6).
      01   DIRA-TITL1  REDEFINES  DIRA.
           02 FILLER      PICTURE     X(240).
           02 INDX-TITL2      PICTURE      9(6)  COMPUTATIONAL.
      01   DIRA-TITL2  REDEFINES  DIRA-TITL1.
           02 INDX-ACCN-T      PICTURE      9(6)  COMPUTATIONAL.
           02 INDX-NEXT-R      PICTURE      9(6)  COMPUTATIONAL.
      01   MESAGE.
           02 FILLER      PICTURE  X(10).
           02 RTRN-INDIC      PICTURE      9(3)     COMPUTATIONAL.
```

```
    02 COUNT-SAVE     PICTURE     9(6)  COMPUTATIONAL.
    02 FILLER  PICTURE    X(4).
    02 FILLER.
      03 PTARAY1     PICTURE  9(6)  COMPUTATIONAL OCCURS 114.
    02 LINK.
        03 BLK-LEN      PICTURE      9(4)  COMPUTATIONAL.
      03 LNK-NO      PICTURE      9(4)  COMPUTATIONAL.
01  RTLIST       PICTURE   9(6)  COMPUTATIONAL.
01  SWIT         PICTURE   9(8)  COMPUTATIONAL.
01  REG8.
    02  FILLER  PICTURE   X(4).
    02  RORW    PICTURE   9(6)  COMPUTATIONAL.
    02  LENG    PICTURE   9(6)  COMPUTATIONAL.
PROCEDURE DIVISION.
START.
    ENTER LINKAGE.
    ENTRY 'MOD4'  USING REG8 RTLIST MESAGE SCOP DIRA SWIT.
    ENTER COBOL.
    MOVE ZERO TO BLK-LEN.
    MOVE LL1 TO L1 SYMBO-KEY.
    IF LEGTH NOT GREATER '0240' GO TO RTRVL.
ERR.
    MOVE 400 TO GOSYLENT LENG.
    ENTER LINKAGE.
    CALL 'GOSYM'  USING L1  GOSYLENT.
    ENTER COBOL.
    MOVE SCOPE TO SCOP.
    MOVE 8 TO RORW.   MOVE 33554456  TO SWIT.
R1. ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
RTRVL.
    MOVE 1 TO RLDARN.
    ENTER LINKAGE.
    CALL  'IOENQ' USING RLDARN RTLIST DIRA SYMBO-KEY.
    ENTER COBOL.
```

```
         IF HIER-COD EQUAL TO '1'  MOVE
         'NO RECORD CAN BE FOUND IN THE FILE BASEDON THE ABOVE DATA. P
-        'LEASE RE-ENTER TITLE-NO MORE THAN 240 CHARACTERS. OR /STOP.'
               TO L2   GO TO ERR.
         MOVE INDX-TITL1 TO REL-KEY SAV.  MOVE  6  TO  RLDARN.
         ENTER LINKAGE.
         CALL 'IOENQ'  USING RLDARN RTLIST  DIRA-TITL1 REL-KEY.
         ENTER COBOL.
         MOVE ZERO TO RTRN-INDIC COUNT-SAVE LNK-NO.
         MOVE INDX-TITL2 TO REL-KEY.
     LB.   MOVE 7 TO RLDARN.
         ENTER LINKAGE.
         CALL 'IOENQ'  USING RLDARN RTLIST  DIRA-TITL2 REL-KEY.
         ENTER COBOL.
         ADD 1 TO BLK-LEN.
         ADD 1 TO COUNT-SAVE.
         MOVE INDX-ACCN-T TO PTARAY1 (COUNT-SAVE).
         IF INDX-NEXT-R = 0
              MOVE 16 TO RORW  MOVE 67108873 TO SWIT  GO TO R1.
         MOVE INDX-NEXT-R TO REL-KEY.   GO TO LB.
```

```
001000 IDENTIFICATION DIVISION.
001010 PROGRAM-ID. 'M5'.
001020 AUTHOR. 'E.S.DAHL'.
000010 REMARKS.                                                         ABSTR
000020     ENTRY POINTS.                                                ABSTR
000020         'MOD5'.                                                  ABSTR
000030                                                                  ABSTR
000030     ACCESS.                                                      ABSTR
000030         SIMPLE SEARCH.                                           ABSTR
000030             SUBJECT.                                             ABSTR
000030                                                                  ABSTR
000050     INVOKED BY.                                                  ABSTR
000060         MODULE 2,                                                ABSTR
000070         MODULE 20.                                               ABSTR
000080                                                                  ABSTR
000090     INVOKES.                                                     ABSTR
000100         MODULE 5 -                                               ABSTR
000110             TO PERMIT NEW SUBJECT REQUEST IF NO DATA FOUND,      ABSTR
000140         MODULE 10 -                                              ABSTR
000130             IF NOT MORE THAN THREE CITATIONS FOUND,              ABSTR
000  0     MODULE 16 -                                                  ABSTR
000150             IF MORE THAN THREE CITATIONS FOUND.                  ABSTR
000160                                                                  ABSTR
000170     SUBROUTINES.                                                 ABSTR
000180         IOENQ,                                                   ABSTR
000190         GOSYM.                                                   ABSTR
000200                                                                  ABSTR
000210     I/O ROUTINES.                                                ABSTR
000220         SUBSEQ  (2),                                             ABSTR
000230         SU1FOR  (8),                                             ABSTR
000240         SU2FOR  (9),                                             ABSTR
000250         WTSCRT (17).                                             ABSTR
000260                                                                  ABSTR
000270     INPUT FILES.                                                 ABSTR
000280         SUBJECT  (COB-ISAM),                                     ABSTR
000290         SUBJECT2 (FORT-REL),                                     ABSTR
```

```
000300          SUBJECT1 (FORT-REL),                                         ABSTR
000310                                                                       ABSTR
000320      OUTPUT FILES.                                                    ABSTR
000330          SCRATCH  (FORT-REL).                                         ABSTR
000340                                                                       ABSTR
000350      MESSAGES.                                                        ABSTR
000370          'THERE IS NO INFORMATION ON FILE FOR          '-             ABSTR
000380          'YOUR REQUEST, TRY AGAIN                      ',             ABSTR
000390              ENTER ANOTHER SUBJECT OR '/STOP',                        ABSTR
000410          'YOUR RESPONSE IS INCORRECT, TRY AGAIN        ',             ABSTR
000410              INPUT LENGTH WAS GREATER THAN 80,                        ABSTR
000420              ENTER SUBJECT NAME CORRECTLY.                            ABSTR
000430                                                                       ABSTR
000440      FUNCTION,                                                        ABSTR
000450          MODULE 5 RECEIVES THE DESIRED SUBJECT NAME WHICH WAS SUB-    ABSTR
000460      MITTED IN RESPONSE TO MODULE 2.,  AN ERROR MESSAGE WITH RETRY    ABSTR
000470      IS ISSUED IF THE NAME GIVEN WAS  NOT  OF  VALID LENGTH.  THE    ABSTR
000480      NAME GIVEN IS  USED  TO  ACCESS THE  'SUBJECT FILE',  WHERE A    BSTR
000490      POINTER IS PICKED UP TO ACCESS THE 'SUBJECT1 FILE',  WHICH IN    BSTR
000500      TURN PROVIDES  A  POINTER TO THE FIRST RELEVANT RECORD IN THE    BSTR
000051      'SUBJECT2 FILE'.   THIS  FILE PROVIDES  ONE  POINTER  TO  THE    ABSTR
000521      'ACCESSION FILE'  AND ANOTHER  TO  THE NEXT RELATIVE RECORD.,    ABSTR
000531      THE  ACCESSION  POINTERS  ARE  STORED  AND  THE  NEXT  RECORD    ABSTR
000541      POINTERS ARE  USED  TO  CHAIN THROUGH THE FILE UNTIL  A  NEXT    ABSTR
000551      RECORD  POINTER  IS  ENCOUNTERED  WITH   A   VALUE  OF  ZERO,    ABSTR
000561      INDICATING AN END OF DATA FOR THIS SUBJECT.   STORAGE OF  THE    ABSTR
000571      ACCESSION POINTERS IS  BY  ARRAYS OF 114 POINTERS EACH, WHICH    ABSTR
000581      ARE WRITTEN TO THE 'SCRATCH FILE',  EXCEPT THE LAST (OR ONLY)    ABSTR
000591      SUCH ARRAY,  WHICH IS LEFT IN THE 'MESAGE'  AREA.,  THIS AREA    ABSTR
000601      ALSO CONTAINS A COUNT OF THE NUMBER OF REFERENCES ACCRETED, A    ABSTR
000611      COUNT OF  THE REFERENCES IN  THE LAST ARRAY, AND  A  CHAINING    ABSTR
000621      POINTER FOR THE 'SCRATCH FILE'.   IF NO REFERENCES ACCRETE TO    ABSTR
000631      A  GIVEN SUBJECT,  A  MESSAGE TO  THIS EFFECT IS  ISSUED WITH    ABSTR
000641      RETRY.                                                           ABSTR
001030 ENVIRONMENT DIVISION.
001032 CONFIGURATION SECTION.
```

```
001040 SOURCE-COMPUTER. IBM-360.
001050 SOURCE-COMPUTER. IBM-360.
001060 DATA DIVISION.
001070 WORKING-STORAGE SECTION.
001080 77  INDX                 PICTURE 9(8)    COMPUTATIONAL.
001090 77  FILLER               PICTURE 9(8)    COMPUTATIONAL.
001100 01  RLDARN                               COMPUTATIONAL.
001110     02  SUBSEQ           PICTURE 9(8)    VALUE   2.
001120     02  SU1FOR           PICTURE 9(8)    VALUE   8.
001130     02  SU2FOR           PICTURE 9(8)    VALUE   9.
001140     02  WTSCRT           PICTURE 9(8)    VALUE   17.
002010 LINKAGE SECTION.
002020 77  RTLIST               PICTURE 9(8)    COMPUTATIONAL.
002030 77  SWIT                 PICTURE 9(8)    COMPUTATIONAL.
002050 01  R8                                   COMPUTATIONAL.
002060     02  FILLER           PICTURE 9(8).
002070     02  RORW             PICTURE 9(8).
002080     02  LEN              PICTURE 9(8).
002090 01  MESAGE.
002100     02  IDNO             PICTURE X(10).
002110     02  CNTRS                            COMPUTATIONAL.
002120         04  FILLER       PICTURE 9(4).
002140         04  TOTAL        PICTURE 9(8).
002130         04  FILLER       PICTURE 9(8).
002150     02  ARRAY.
002160         04  PTR          PICTURE 9(8)    COMPUTATIONAL. OCCURS 114.
002170     02  LINK                             COMPUTATIONAL.
002180         04  BLK-LEN      PICTURE 9(4).
002190         04  LNK-NO       PICTURE 9(4).
002200 01  SCOPE.
002210     02  IN-LEN           PICTURE 9(8)    COMPUTATIONAL.
002220     02  TUBE             PICTURE X(80).
002230     02  FILLER           PICTURE X(400).
003010 01  SUBJECT.
003020     02  ERRBYTE          PICTURE X.
003030         88  NOHIT                        VALUE   '1'.
```

```
003040      02  FILLER      PICTURE X(80).
003050      02  SUBJ1-INDX  PICTURE 9(6).
003060      02  FILLER      PICTURE X(407).
003070 01   SUBJ1 REDEFINES SUBJECT.
003080      02  FILLER      PICTURE X(80).
003090      02  SUBJ2-INDX  PICTURE 9(8)    COMPUTATIONAL.
003100      02  FILLER      PICTURE X(10).
003110      02  FILLER      PICTURE X(398).
003120 01   SUBJ2 REDEFINES SUBJECT.
003130      02  FLR                         COMPUTATIONAL.
003140      04  ACCN-INDX   PICTURE 9(8).
003150      04  NEXT-INDX   PICTURE 9(8).
003160      04  FILLER      PICTURE 9(8).
003170      04  FILLER      PICTURE 9(8).
003180      02  FILLER      PICTURE X(476).
004010 PROCEDURE DIVISION.
004020      ENTER LINKAGE.
004030      ENTRY 'MOD5' USING R8, RTLIST, MESAGE,
004040                      SCOPE, SUBJECT, SWIT.
004050      ENTER COBOL.
004060 I1.  MOVE 1 TO BLK-LEN.
004070 I2.  MOVE 0 TO PTR (BLK-LEN).
004080      IF BLK-LEN = 114 NEXT SENTENCE ELSE
004090      ADD 1 TO BLK-LEN GO TO I2.
004092 I4.  EXIT.
004100 I5.  PERFORM I1 MOVE 0 TO LNK-NO, TOTAL.
004110      IF IN-LEN   80 GO TO BAD-RQST.
004120      IF IN-LEN ) 1  GO TO BAD-RQST.
004130 GET-ISAM.
004140      ENTER LINKAGE.
004150      CALL 'IOENQ' USING SUBSEQ, RTLIST, SUBJECT, TUBE.
004160          ENTER COBOL.
004170      IF NOHIT GO TO NO-DATA.
004180      MOVE SUBJ1-INDX TO INDX.
004190 GET-SUBJ1.
004200      ENTER LINKAGE.
```

```
004210      CALL 'IOENQ' USING SU1FOR, RTLIST, SUBJ1, INDX.
004220      ENTER COBOL.
004230      MOVE SUBJ2-INDX TO INDX.
005010 GET-SUBJ2.
005020      ENTER LINKAGE.
005030      CALL 'IOENQ' USING SU2FOR, RTLIST, SUBJ2, INDX.
005040      ENTER COBOL.
005050 PROCESS-REC2.
005060      ADD 1 TO TOTAL.
005070      MOVE ACCN-INDX TO PTR (BLK-LEN).
005080      IF NEXT-INDX = 0 GO TO EOD.
005090 P2. IF BLK-LEN NOT = 114 ADD 1 TO BLK-LEN,
005100      MOVE NEXT-INDX TO INDX, GO TO GET-SUBJ2.
005110 PUT-SCRX.
005120      ENTER LINKAGE.
005130      CALL 'IOENQ' USING WTSCRT, RTLIST, ARRAY, LINK.
005140      ENTER COBOL.
005142      MOVE 1 TO BLK-LEN PERFORM I2 THRU I4.
005150      MOVE 0 TO BLK-LEN GO TO P2.
005160 EOD.
005170      IF TOTAL = 0 GO TO NO-DATA.
005180      IF TOTAL   3 GO TO MOD16.
005190 MOD10.
005200      MOVE 16 TO RORW.
005210      MOVE 67108873 TO SWIT.
005220      GO TO RTRN.
006010 MOD16.
006020      MOVE 16 TO RORW.
006030      MOVE 67108879 TO SWIT.
006040 RTRN.
006050      ENTER LINKAGE.
006060      RETURN.
006070      ENTER COBOL.
006080 NO-DATA.
006090      MOVE                      'THERE IS NO INFORMATION ON FILE FOR
006100-                                'YOUR REQUEST, TRY AGAIN'
```

```
006110          TO TUBE.
006120          MOVE 80 TO LEN, GO TO PUT-SCOPE.
006130     BAD-RQST.
006140          MOVE 'YOUR RESPONSE IS INCORRECT-TRY AGAIN' TO TUBE.
006150          MOVE 40 TO LEN.
006160     PUT-SCOPE.
006170          MOVE 8 TO RORW.
006180          MOVE 33554436 TO SWIT.
006190          ENTER LINKAGE.
006200          CALL 'GOSYM' USING TUBE, LEN.
006210          ENTER COBOL.
006220          GO TO RTRN.
```

```
IDENTIFICATION DIVISION.
PROGRAM-ID. 'M10'.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-360.
OBJECT-COMPUTER. IBM-360.
DATA DIVISION.
WORKING-STORAGE SECTION.
77  SUB                     PICTURE 9(3) VALUE 0.
77  RLDARN                  PICTURE 9(6) COMPUTATIONAL.
77  FORT-KEY                PICTURE 9(6) COMPUTATIONAL.
77  AU-KEY-HOLD             PICTURE 9(6) COMPUTATIONAL.
77  ACN-HOLD                PICTURE X(16).
77  CT                      PICTURE 9(3) VALUE 1.
77  CTR                     PICTURE 9(3) VALUE 1.
77  TALLI                   PICTURE 9(3) VALUE 0.
77  DESC-SYM-KEY            PICTURE X(16) VALUE SPACES.
77  NTFR2                   PICTURE 9(6) COMPUTATIONAL.
77  NTT-HOLD                PICTURE 9(6) COMPUTATIONAL.
77  NTS-HOLD                PICTURE 9(6) COMPUTATIONAL.
77  NTOA-HOLD               PICTURE 9(6) COMPUTATIONAL.
77  GORK                    PICTURE 9(6) COMPUTATIONAL.
77  RTRN-NDIC               PICTURE 9(3) COMPUTATIONAL.
77  PUB-HOLD                PICTURE 9(6).
77  PLACE-HOLD              PICTURE 9(6).
01  HOLD-ARAY.
    02  HA                  PICTURE X(39) OCCURS 8 TIMES.
01  LYNE.
    02  LN                  PICTURE X OCCURS 39 TIMES.
01  HOLDER.
    02  H                   PICTURE X OCCURS 39 TIMES.
LINKAGE SECTION.
77  RTLIST                  PICTURE 9(6) COMPUTATIONAL.
77  SWIT                    PICTURE 9(8) COMPUTATIONAL.
01  REG8.
    02  FILLER              PICTURE X(4).
```

```
           02  RORW                    PICTURE 9(6) COMPUTATIONAL.
           02  LENG                    PICTURE 9(6) COMPUTATIONAL.
       01  MESAGE.
           02  ID-NO                   PICTURE X(10).
           02  RTRN-INDIC              PICTURE 9(3) COMPUTATIONAL.
           02  TOTAL-COUNT             PICTURE 9(5) COMPUTATIONAL.
           02  ALSO-COUNT              PICTURE 9(6) COMPUTATIONAL.
           02  POINTER.
               03  PTARAY              PICTURE 9(6) COMPUTATIONAL OCCURS 114
      -    TIMES.
           02  FOR-KEY.
               03  BLK-LEN             PICTURE 9(3) COMPUTATIONAL.
               03  LINK-NO             PICTURE 9(3) COMPUTATIONAL.
       01  SCOPE.
           02  MES-LENG                PICTURE 9(5) COMPUTATIONAL.
           02  LYNES.
               03  L                   PICTURE X(40) OCCURS 12 TIMES.
       01  DIRA.
           02  ACCN-NR                 PICTURE X(16).
           02  NDEX-T-TITL             PICTURE 9(6) COMPUTATIONAL.
           02  NDEX-T-SUBJ             PICTURE 9(6) COMPUTATIONAL.
           02  NDEX-T-AUTHOR           PICTURE 9(6) COMPUTATIONAL.
           02  NDEX-T-O-AUTHOR         PICTURE 9(6) COMPUTATIONAL.
           02  A-LEV                   PICTURE 9(6) COMPUTATIONAL.
           02  NDEX-T-N-ACCN           PICTURE 9(6) COMPUTATIONAL.
           02  FILLER                  PICTURE X(432).
       01  DIRA1 REDEFINES DIRA.
           02  AUTHOR                  PICTURE X(213).
           02  NDEX-T-N-AUTH           PICTURE 9(5) COMPUTATIONAL.
           02  NDEX-T-ACCN-A           PICTURE 9(5) COMPUTATIONAL.
           02  SEE-NDEX                PICTURE 9(5) COMPUTATIONAL.
           02  YOB                     PICTURE X(8).
           02  FILLER                  PICTURE X(248).
       01  DIRA2 REDEFINES DIRA1.
           02  TITLE                   PICTURE X(240).
           02  NDEX-T-FRST-REC2        PICTURE 9(6) COMPUTATIONAL.
```

```
    02  FILLER              PICTURE X(238).
01  DIRA3 REDEFINES DIRA2.
    02  NDEX-T-ACCN-T       PICTURE 9(6) COMPUTATIONAL.
    02  NDEX-T-NEXT-REC2    PICTURE 9(6) COMPUTATIONAL.
    02  NDEX-T-O-TITL       PICTURE 9(6) COMPUTATIONAL.
    02  FILLER              PICTURE X(466).
01  DIRA4 REDEFINES DIRA3.
    02  SUBJECT             PICTURE X(80).
    02  NDEX-T-FRST-SUB-REC2 PICTURE 9(6) COMPUTATIONAL.
    02  MESH-NR             PICTURE X(10).
    02  FILLER              PICTURE X(388).
01  DIRA5 REDEFINES DIRA4.
    02  NDEX-T-ACCN-S       PICTURE 9(6) COMPUTATIONAL.
    02  NDEX-T-NEXT-RECORD2 PICTURE 9(6) COMPUTATIONAL.
    02  NDEX-T-O-SUBJ       PICTURE 9(6) COMPUTATIONAL.
    02  PAGE-NR             PICTURE 9(6) COMPUTATIONAL.
    02  FILLER              PICTURE X(460).
01  DIRA6 REDEFINES DIRA5.
    02  AU-KEY-HOLDER       PICTURE 9(6) COMPUTATIONAL.
    02  ACN-HOLDER          PICTURE X(16).
    02  NTT-HOLDER          PICTURE 9(6) COMPUTATIONAL.
    02  NTS-HOLDER          PICTURE 9(6) COMPUTATIONAL.
    02  NTOA-HOLDER         PICTURE 9(6) COMPUTATIONAL.
    02  PUB-HOLDER          PICTURE 9(6).
    02  PLACE-HOLDER        PICTURE 9(6).
    02  NTFR3               PICTURE 9(6) COMPUTATIONAL.
    02  FORT-H1             PICTURE 9(6) COMPUTATIONAL.
    02  FILLER              PICTURE X(110).
    02  HOLD-ARAYD          PICTURE X(312).
01  DIRA7 REDEFINES DIRA6.
    02  FILLER              PICTURE X(76).
    02  CLASS               PICTURE X(3).
    02  NUMBER              PICTURE X(4).
    02  CUTTER-1            PICTURE X(2).
    02  CUTTER-2            PICTURE X(2).
    02  YEAR                PICTURE X(3).
    02  FILLER                  PICTURE X(39).
    02  NOTES                   PICTURE X(284).
    02  FILLER                  PICTURE X(22).
    02  PLACE-CODE              PICTURE 9(6).
    02  PUB-CODE                PICTURE 9(6).
    02  FILLER                  PICTURE X(3).
    02  COLAT                   PICTURE X(40).
01  DIRA8 REDEFINES DIRA7.
    02  PUBLISHER               PICTURE X(68).
    02  FILLER                  PICTURE X(416).
01  DIRA9 REDEFINES DIRA8.
    02  LOCATION                PICTURE X(120).
    02  FILLER                  PICTURE X(364).
PROCEDURE DIVISION.
    ENTER LINKAGE.
    ENTRY 'MOD10' USING REG8 RTLIST MESAGE SCOPE DIRA SWIT.
    ENTER COBOL.
    MOVE SPACES TO LYNE.
    MOVE NTOA-HOLDER TO NTOA-HOLD.
    MOVE ACN-HOLDER TO ACN-HOLD.
    MOVE PLACE-HOLDER TO PLACE-HOLD.
    MOVE PUB-HOLDER TO PUB-HOLD.
    MOVE AU-KEY-HOLDER TO AU-KEY-HOLD.
    MOVE NTS-HOLDER TO NTS-HOLD.
    MOVE NTT-HOLDER TO NTT-HOLD.
    MOVE NTFR3 TO NTFR2.
    MOVE RTRN-INDIC TO RTRN-NDIC.
        MOVE BLK-LEN TO SUB.
    IF RTRN-NDIC   0 GO TO ID-SEE.
BEGINNING.
    IF BLK-LEN   0 MOVE BLK-LEN TO SUB MOVE PTARAY (SUB)
    TO FORT-KEY GO TO CALIT1.
    IF TOTAL-COUNT = 0 GO TO SC-OUT1.
    MOVE 16 TO RORW.
    MOVE 33554443 TO SWIT.
    ENTER LINKAGE.
```

```
          RETURN.
          ENTER COBOL.
      SC-OUT1.
          MOVE 'THERE ARE NO MORE CITATIONS ON FILE RELATIVE TO YOUR
          ' REQUEST                        *' TO LYNES.
          ENTER LINKAGE.
          CALL 'CARRET' USING SCOPE.
          ENTER COBOL.
          MOVE 8 TO RORW.
          MOVE 80 TO LENG.
          MOVE 16777314 TO SWIT.
          ENTER LINKAGE.
          RETURN.
          ENTER COBOL.
      CALIT1.
          MOVE 4 TO RLDARN.
          MOVE SPACES TO DIRA.
          ENTER LINKAGE.
          CALL 'IOENQ' USING RLDARN RTLIST DIRA FORT-KEY.
          ENTER COBOL.
      GORKY.
          MOVE SPACES TO LYNE HOLDER.
          MOVE 0 TO AU-KEY-HOLD.
          MOVE 0 TO NTT-HOLD.
          MOVE 0 TO NTS-HOLD.
          MOVE 0 TO PUB-HOLD.
          MOVE 0 TO NTOA-HOLD.
          MOVE 0 TO PLACE-HOLD.
          MOVE 0 TO NTFR2.
          MOVE SPACES TO ACN-HOLD.
          MOVE NDEX-T-AUTHOR TO AU-KEY-HOLD.
          MOVE NDEX-T-O-AUTHOR TO NTOA-HOLD.
          MOVE ACCN-NR TO ACN-HOLD.
          IF NDEX-T-TITL IS NOT EQUAL TO 0 MOVE NDEX-T-TITL TO NTT-HOLD
          .
          IF NDEX-T-SUBJ IS NOT EQUAL TO 0 MOVE NDEX-T-SUBJ TO NTS-HOLD
```

```
-     GO TO CALIT2.
     IF NTOA-HOLD = AU-KEY-HOLD GO TO CALIT2.
     MOVE NTOA-HOLD TO FORT-KEY.
     MOVE 5 TO RLDARN.
     MOVE SPACES TO DIRA.
     ENTER LINKAGE.
     CALL 'IOENQ' USING RLDARN RTLIST DIRA FORT-KEY.
     ENTER COBOL.
     MOVE NDEX-T-ACCN-A TO FORT-KEY.
     MOVE NDEX-T-ACCN-A TO PTARAY (SUB).
     MOVE 4 TO RLDARN.
     MOVE SPACES TO DIRA.
     ENTER LINKAGE.
     CALL 'IOENQ' USING RLDARN RTLIST DIRA FORT-KEY.
     ENTER COBOL.
     MOVE NDEX-T-AUTHOR TO GORK.
     MOVE NDEX-T-O-AUTHOR TO NDEX-T-AUTHOR.
     MOVE GORK TO NDEX-T-O-AUTHOR.
     GO TO GORKY.
 CALIT2.
     MOVE ACN-HOLD TO DESC-SYM-KEY.
     MOVE 3 TO RLDARN.
     MOVE SPACES TO DIRA.
     ENTER LINKAGE.
     CALL 'IOENQ' USING RLDARN RTLIST DIRA DESC-SYM-KEY.
     ENTER COBOL.
     MOVE PLACE-CODE TO PLACE-HOLD.
     MOVE PUB-CODE TO PUB-HOLD.
     MOVE SPACES TO HOLDER.
     MOVE CLASS TO HOLDER.
 CLAS-MOVE.
     MOVE H (CT) TO LN (CTR).
     ADD 1 TO CT.
     ADD 1 TO CTR.
     IF CT IS NOT   3 GO TO CLAS-MOVE.
     MOVE 1 TO CT.
```

```
        MOVE '/' TO LN (CTR).
        ADD 1 TO CTR.
        MOVE SPACES TO HOLDER.
        MOVE NUMBER TO HOLDER.
    NUM-MOVE.
        MOVE H (CT) TO LN (CTR).
        ADD 1 TO CT.
        ADD 1 TO CTR.
        IF CT IS NOT   4 GO TO NUM-MOVE.
        MOVE 1 TO CT.
        MOVE '/' TO LN (CTR).
        ADD 1 TO CTR.
        MOVE SPACES TO HOLDER.
        MOVE CUTTER-1 TO HOLDER.
    CT-1-MOVE.
        MOVE H (CT) TO LN (CTR).
        ADD 1 TO CT.
        ADD 1 TO CTR.
        IF CT IS NOT   2 GO TO CT-1-MOVE.
        MOVE 1 TO CT.
        MOVE SPACES TO HOLDER.
        MOVE CUTTER-2 TO HOLDER.
    CT-2-MOVE.
        MOVE H (CT) TO LN (CTR).
        ADD 1 TO CT.
        ADD 1 TO CTR.
        IF CT IS NOT   2 GO TO CT-2-MOVE.
        MOVE 1 TO CT.
        MOVE '/' TO LN (CTR).
        ADD 1 TO CTR.
        MOVE SPACES TO HOLDER.
        MOVE YEAR TO HOLDER.
    YEAR-MOVE.
        MOVE H (CT) TO LN (CTR).
        ADD 1 TO CT.
        ADD 1 TO CTR.
```

```
     IF CT IS NOT   3 GO TO YEAR-MOVE.
 ID-SEE.
     MOVE 1 TO CT.
     MOVE 30 TO CTR.
     MOVE SPACES TO HOLDER.
     MOVE ID-NO TO HOLDER.
 ID-MOVE.
     MOVE H (CT) TO LN (CTR).
     ADD 1 TO CT.
     ADD 1 TO CTR.
     IF CT IS NOT   10 GO TO ID-MOVE.
     MOVE SPACES TO LYNES.
     MOVE LYNE TO L (1).
     MOVE 1 TO CT.
     MOVE 2 TO CTR.
     IF RTRN-NDIC = 1 MOVE HOLD-ARAYD TO HOLD-ARAY GO TO
     SPAS-CHEK2.
     IF RTRN-NDIC = 2 MOVE HOLD-ARAYD TO HOLD-ARAY GO TO
     SPAS-CHEK3.
     IF RTRN-NDIC = 3 MOVE HOLD-ARAYD TO HOLD-ARAY GO TO
     SPAS-CHEK4.
     IF RTRN-NDIC = 5 MOVE HOLD-ARAYD TO HOLD-ARAY GO TO
     SPAS-CHEK6.
     IF RTRN-NDIC = 6 MOVE FORT-H1 TO NDEX-T-FRST-SUB-REC2 MOVE
     HOLD-ARAYD TO HOLD-ARAY GO TO SPAS-CHEK7.
     IF RTRN-NDIC = 7 MOVE FORT-H1 TO NDEX-T-ACCN-A MOVE
     HOLD-ARAYD TO HOLD-ARAY GO TO SPAS-CHEK8.
     IF RTRN-NDIC = 8 MOVE HOLD-ARAYD TO HOLD-ARAY GO TO
     SPAS-CHEK9.
     MOVE AU-KEY-HOLD TO FORT-KEY.
     MOVE 5 TO RLDARN.
     MOVE SPACES TO DIRA.
     ENTER LINKAGE.
     CALL 'IDENQ' USING RLDARN RTLIST DIRA FORT-KEY.
     ENTER COBOL.
     MOVE SPACES TO HOLD-ARAY.
```

```
    MOVE AUTHOR TO HOLD-ARAY.
SPAS-CHEK1.
    IF HA (CT) IS NOT EQUAL TO SPACES ADD 1 TO CT ADD 1 TO CTR
    GO TO SPAS-CHEK1.
    MOVE CTR TO TALLI.
AUTH-MOVE.
    SUBTRACT 1 FROM CTR.
    SUBTRACT 1 FROM CT.
    MOVE HA (CT) TO L (CTR).
    IF CT   1 GO TO AUTH-MOVE.
    IF NTT-HOLD = 0 GO TO PUBL-SEE.
    MOVE NTT-HOLD TO FORT-KEY.
    MOVE 6 TO RLDARN.
    MOVE SPACES TO DIRA.
    ENTER LINKAGE.
    CALL 'IOENQ' USING RLDARN RTLIST DIRA FORT-KEY.
    ENTER COBOL.
    MOVE TALLI TO CTR.
    MOVE SPACES TO HOLD-ARAY.
    MOVE TITLE TO HOLD-ARAY.
    MOVE NDEX-T-FRST-REC2 TO NTFR2.
SPAS-CHEK2.
    IF HA (CT) IS NOT EQUAL TO SPACES ADD 1 TO CT ADD 1 TO CTR
    GO TO SPAS-CHEK2.
    MOVE CTR TO TALLI.
TITL-MOVE.
    IF TALLI   13 MOVE HOLD-ARAY TO HOLD-ARAYD MOVE 1 TO
    RTRN-NDIC GO TO PREP-SCOPE2.
    SUBTRACT 1 FROM CT.
    SUBTRACT 1 FROM CTR.
    MOVE HA (CT) TO L (CTR).
    IF CT   1 GO TO TITL-MOVE.
PUBL-SEE.
    MOVE PUB-HOLD TO FORT-KEY.
    IF FORT-KEY = 0 GO TO SPAS-CHEK3.
    MOVE 12 TO RLDARN.
    MOVE SPACES TO DIRA.
    ENTER LINKAGE.
    CALL 'IDENQ' USING RLDARN RTLIST DIRA FORT-KEY.
    ENTER COBOL.
    MOVE TALLI TO CTR.
    MOVE SPACES TO HOLD-ARAY.
    MOVE PUBLISHER TO HOLD-ARAY.
SPAS-CHEK3.
    IF HA (CT) IS NOT EQUAL TO SPACES ADD 1 TO CT ADD 1 TO CTR
    GO TO SPAS-CHEK3.
    MOVE CTR TO TALLI.
PUB-MOVE.
    IF TALLI   13 MOVE HOLD-ARAY TO HOLD-ARAYD MOVE 2 TO
    RTRN-NDIC GO TO PREP-SCOPE2.
    SUBTRACT 1 FROM CT.
    SUBTRACT 1 FROM CTR.
    MOVE HA (CT) TO L (CTR).
    IF CT   1 GO TO PUB-MOVE.
    MOVE PLACE-HOLD TO FORT-KEY.
    IF FORT-KEY = 0 GO TO SPAS-CHEK4.
    MOVE 11 TO RLDARN.
    MOVE SPACES TO DIRA.
    ENTER LINKAGE.
    CALL 'IDENQ' USING RLDARN RTLIST DIRA FORT-KEY.
    ENTER COBOL.
    MOVE TALLI TO CTR.
    MOVE SPACES TO HOLD-ARAY.
    MOVE LOCATION TO HOLD-ARAY.
SPAS-CHEK4.
    IF HA (CT) IS NOT EQUAL TO SPACES ADD 1 TO CT ADD 1 TO CTR
    GO TO SPAS-CHEK4.
    MOVE CTR TO TALLI.
    IF TALLI   13 MOVE HOLD-ARAY TO HOLD-ARAYD MOVE 3 TO
    RTRN-NDIC GO TO PREP-SCOPE2.
LOC-MOVE.
    SUBTRACT 1 FROM CT.
    SUBTRACT 1 FROM CTR.
    MOVE HA (CT) TO L (CTR).
    IF CT   1 GO TO LOC-MOVE.
    MOVE ACN-HOLD TO DESC-SYM-KEY.
    MOVE 3 TO RLDARN.
    MOVE SPACES TO DIRA.
    ENTER LINKAGE.
    CALL 'IOENQ' USING RLDARN RTLIST DIRA DESC-SYM-KEY.
    ENTER COBOL.
    MOVE COLAT TO L (TALLI).
    ADD 1 TO TALLI.
    IF NOTES = SPACES GO TO SUBJ-SEE.
    MOVE TALLI TO CTR.
    MOVE SPACES TO HOLD-ARAY.
    MOVE NOTES TO HOLD-ARAY.
SPAS-CHEK6.
    IF HA (CT) IS NOT EQUAL TO SPACES ADD 1 TO CT ADD 1 TO CTR
    GO TO SPAS-CHEK6.
    MOVE CTR TO TALLI.
    IF TALLI   13 MOVE HOLD-ARAY TO HOLD-ARAYD MOVE 5 TO
    RTRN-NDIC GO TO PREP-SCOPE2.
NOTES-MOVE.
    SUBTRACT 1 FROM CT.
    SUBTRACT 1 FROM CTR.
    MOVE HA (CT) TO L (CTR).
    IF CT   1 GO TO NOTES-MOVE.
SUBJ-SEE.
    IF NTS-HOLD = 0 GO TO AUTH2-SEE.
    MOVE NTS-HOLD TO FORT-KEY.
THRU-AGAIN.
    MOVE 8 TO RLDARN.
    MOVE SPACES TO DIRA.
    ENTER LINKAGE.
    CALL 'IOENQ' USING RLDARN RTLIST DIRA FORT-KEY.
    ENTER COBOL.
    MOVE TALLI TO CTR.
```

```
     MOVE SPACES TO HOLD-ARAY.
     MOVE SUBJECT TO HOLD-ARAY.
     MOVE 1 TO CT.
 SPAS-CHEK7.
     IF HA (CT) IS NOT EQUAL TO SPACES ADD 1 TO CT ADD 1 TO CTR
     GO TO SPAS-CHEK7.
     MOVE CTR TO TALLI.
     IF TALLI   13 MOVE 6 TO RTRN-NDIC MOVE HOLD-ARAY TO
     HOLD-ARAYD MOVE NDEX-T-FRST-SUB-REC2 TO FORT-H1 GO TO
     PREP-SCOPE2.
 SUBJ-MOVE.
     SUBTRACT 1 FROM CT.
     SUBTRACT 1 FROM CTR.
     MOVE HA (CT) TO L (CTR).
     IF CT   1 GO TO SUBJ-MOVE.
 SUB2-SEE.
     MOVE NDEX-T-FRST-SUB-REC2 TO FORT-KEY.
     MOVE 9 TO RLDARN.
 LOOK-AGAIN.
     MOVE SPACES TO DIRA.
     ENTER LINKAGE.
     CALL 'IDENQ' USING RLDARN RTLIST DIRA FORT-KEY.
     ENTER COBOL.
     IF NDEX-T-ACCN-S IS NOT EQUAL TO PTARAY (SUB) MOVE
     NDEX-T-NEXT-REC2 TO FORT-KEY GO TO LOOK-AGAIN.
     IF NDEX-T-O-SUBJ = NTS-HOLD GO TO AUTH2-SEE.
     MOVE NDEX-T-O-SUBJ TO FORT-KEY.
     GO TO THRU-AGAIN.
 AUTH2-SEE.
     IF NTOA-HOLD = AU-KEY-HOLD GO TO TITL2-SEE.
     MOVE NTOA-HOLD TO FORT-KEY.
     MOVE 5 TO RLDARN.
     MOVE SPACES TO DIRA.
     ENTER LINKAGE.
     CALL 'IDENQ' USING RLDARN RTLIST DIRA FORT-KEY.
     ENTER COBOL.
```

```
          MOVE TALLI TO CTR.
          MOVE SPACES TO HOLD-ARAY.
          MOVE AUTHOR TO HOLD-ARAY.
      SPAS-CHEK8.
          IF HA (CT) IS NOT EQUAL TO SPACES ADD 1 TO CT ADD 1 TO CTR
          GO TO SPAS-CHEK8.
          MOVE CTR TO TALLI.
          IF TALLI   13 MOVE NDEX-T-ACCN-A TO FORT-H1 MOVE HOLD-ARAY
          TO HOLD-ARAYD MOVE 7 TO RTRN-NDIC GO TO PREP-SCOPE2.
      AUTH1-MOVE.
          SUBTRACT 1 FROM CT.
          SUBTRACT 1 FROM CTR.
          MOVE HA (CT) TO L (CTR).
          IF CT   1 GO TO AUTH1-MOVE.
          MOVE NDEX-T-ACCN-A TO FORT-KEY.
      JONES-2.
          MOVE 4 TO RLDARN.
          MOVE SPACES TO DIRA.
          ENTER LINKAGE.
          CALL 'IDENQ' USING RLDARN RTLIST DIRA FORT-KEY.
          ENTER COBOL ,
          IF ACCN-NR NOT = ACN-HOLD GO TO JONES-1.
          MOVE NDEX-T-O-AUTHOR TO NTOA-HOLD.
          GO TO AUTH2-SEE.
      TITL2-SEE.
          MOVE NTFR2 TO FORT-KEY.
          MOVE 7 TO RLDARN.
      CALIT3.
          MOVE SPACES TO DIRA.
          ENTER LINKAGE.
          CALL 'IDENQ' USING RLDARN RTLIST DIRA FORT-KEY.
          ENTER COBOL.
          IF NDEX-T-ACCN-T IS NOT EQUAL TO PTARAY (SUB) MOVE
          NDEX-T-NEXT-REC2 TO FORT-KEY GO TO CALIT3.
          IF NDEX-T-O-TITL = NTT-HOLD SUBTRACT 1 FROM BLK-LEN
          SUBTRACT 1 FROM TOTAL-COUNT GO TO ENDER1.
```

```
ID-MOVER.
    MOVE H (CT) TO LN (CTR).
    ADD 1 TO CT.
    ADD 1 TO CTR.
    IF CT IS NOT   10 GO TO ID-MOVER.
    MOVE LYNE TO L (TALLI).
SC-OUT2.
    MOVE NTOA-HOLD TO NTOA-HOLDER.
    MOVE ACN-HOLD TO ACN-HOLDER.
    MOVE PLACE-HOLD TO PLACE-HOLDER.
    MOVE PUB-HOLD TO PUB-HOLDER.
    MOVE AU-KEY-HOLD TO AU-KEY-HOLDER.
    MOVE NTS-HOLD TO NTS-HOLDER.
    MOVE NTT-HOLD TO NTT-HOLDER.
    MOVE NTFR2 TO NTFR3.
    MOVE RTRN-NDIC TO RTRN-INDIC.
    ENTER LINKAGE.
    CALL 'CARRET' USING SCOPE.
    ENTER COBOL.
ENDER.
    MOVE 8 TO RORW.
    MOVE 480 TO LENG.
    MOVE 67108873 TO SWIT.
    ENTER LINKAGE.
    RETURN.
    ENTER COBOL.
ENDER1.
    MOVE 0 TO RTRN-INDIC.
    GO TO ENDER.
JONES-1.
    IF NDEX-T-N-ACCN = 0 GO TO TITL2-SEE.
    MOVE NDEX-T-N-ACCN TO FORT-KEY.
    GO TO JONES-2.
    MOVE NDEX-T-O-TITL TO FORT-KEY.
    MOVE 6 TO RLDARN.
    MOVE SPACES TO DIRA.
    ENTER LINKAGE.
    CALL 'IOENQ' USING RLDARN RTLIST DIRA FORT-KEY.
    ENTER COBOL.
    MOVE NDEX-T-FRST-REC2 TO FORT-KEY.
    MOVE TALLI TO CTR.
    MOVE SPACES TO HOLD-ARAY.
    MOVE TITLE TO HOLD-ARAY.
    MOVE 0 TO NTFR2.
    MOVE NDEX-T-FRST-REC2 TO NTFR2.
    MOVE 1 TO CT.
SPAS-CHEK9.
    IF HA (CT) IS NOT EQUAL TO SPACES ADD 1 TO CT ADD 1 TO CTR
    GO TO SPAS-CHEK9.
    MOVE CTR TO TALLI.
TITL2-MOVE.
    IF TALLI   13 MOVE HOLD-ARAY TO HOLD-ARAYD MOVE 8 TO
    RTRN-NDIC GO TO PREP-SCOPE2.
    SUBTRACT 1 FROM CT.
    SUBTRACT 1 FROM CTR.
    MOVE HA (CT) TO L (CTR).
    IF CT   1 GO TO TITL2-MOVE.
    GO TO TITL2-SEE.
PREP-SCOPE2.
    IF RTRN-NDIC IS NOT EQUAL TO 8 GO TO SC-OUT2.
    IF TOTAL-COUNT   1 GO TO SC-OUT2.
LAST-SCOPE.
    IF L (CT) IS NOT EQUAL TO SPACES MOVE L (CT) TO L (CTR)
    ADD 1 TO CT ADD 1 TO CTR GO TO LAST-SCOPE.
    MOVE CT TO TALLI.
    MOVE 15 TO CTR.
    MOVE 1 TO CT.
    MOVE SPACES TO HOLDER.
    MOVE ID-NO TO HOLDER.
```